ENABLING IN SITU & CONTEXT-BASED MOTION GESTURE DESIGN

A Thesis
Presented to
The Academic Faculty

by

Aman Parnami

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in the
School of Interactive Computing

Georgia Institute of Technology
May 2017

ENABLING IN SITU & CONTEXT-BASED MOTION GESTURE DESIGN

Approved By:

Dr. Gregory D. Abowd, Co-Advisor
School of Interactive Computing
Georgia Institute of Technology

Dr. Betsy DiSalvo, Co-Advisor
School of Interactive Computing
Georgia Institute of Technology

Dr. Thad Starner
School of Interactive Computing
Georgia Institute of Technology

Dr. Keith Edwards
School of Interactive Computing
Georgia Institute of Technology

Dr. Björn Hartmann
Electrical Engg. & Computer Science
University of California Berkeley

Dr. Yang Li
Staff Research Scientist
Google Inc.

March 31^{st} 2017

ACKNOWLEDGEMENTS

I thank God for placing me in the company of such an amazing group of people, who believed in my abilities and supported me throughout this journey.

First of all, I will like to thank my academic parents, Gregory D. Abowd and Betsy DiSalvo, for taking me in when I was naïve and letting me out after they had prepared me for the future I had chosen. I am forever grateful to them for giving me their best. Gregory is the reason I got into research and stayed at Georgia Tech to pursue a Ph.D. He showed more faith in me than I did myself. He generously shared his time, provided financial support, and opened his vast network to create the best learning environment I could expect. Only he could top that with an accommodating friendship and playful interactions which I will cherish for the rest of our lives.

While Gregory, figuratively, gave me enough rope to hang myself, Betsy kept me grounded and helped me identify the style of research I had biggest strengths in. Even though I started working with her on a topic in her core research area and soon moved to my thesis topic which was new for her too, she dedicated her efforts to unquestionably support my decision. Betsy opened my mind to the vast field of education research and inspired me with her commitment to empowering the underprivileged. I take pride in being her first Ph.D. student and take comfort in the strength of our relationship.

Alongside my advisors, many mentors motivated me, challenged me, and shared their wisdom with me when I needed it the most. My other committee members Thad Starner, Yang Li, Keith Edwards, and Björn Hartmann expected nothing short of the highest standards they have set in their own exceptional careers. Thad Starner and Yang Li, with their constant involvement, have influenced my work more than I have given them credit for. Rosa Arriaga, Agatha Rozga, Thomaz Ploetz, and Kaya Barbaro filled gaps in my understanding of their domains of expertise. From the School of Industrial Design, Jim Budd and Kevin Shankwiler have instilled in me the significance of good design and taught me the designerly ways of thinking and learning. At Microsoft Research, Mike Sinclair exposed me to the working of an inventor and infused me with the spirit of making. Lastly, Ali Mazalek nurtured my curiosity and guided my energy into the productive endeavor of building interactive artifacts.

My academic family in the Ubicomp, Synlab, and CAT labs and other friends at Georgia Tech have augmented my intellect and contributed to my wellbeing. A shout-out to Edison Thomaz and Caleb Southern for introducing me to research in HCI, to Ramik Sadana for inviting me to Georgia Tech and for being a co-conspirator, to Gabriel Reyes for being a brother to me, and to Yi Han for keeping my spirits high. I will also cherish the memories I have with my other labmates and friends including Susan Robinson, Andy Wu, Ivan Riobo, Ceara Byrne, Firaz Peer, Hwajung Hong, Chad Stopler, Cheng Zhang, Dingtian Zhang, Nivedita Arora, Abdelkareem Bedri, Jung Wook Park, Hayley Evans, Kayla Desportes, Zane Cochran, Joelle Alcaidinho, Giancarlo Valentin and many more. Additionally, I have been fortunate to collaborate with and mentor an awesome pack of students including Apurva Gupta, Pratik Shah, Nick Stoltzfus, James Park, Vishvak Murahari, and Mengyang Shi.

I shall not forget the support many staff members provided me behind the scenes to run the ship smoothly. In particular, I will like to thank Scott Gilliland, Stephanie Tofighi, Jessica Celestine, Cynthia Bryant, Chanel Bridges, and Monica Ross for their assistance.

Sonal Sukheeja, my best friend, my love, and my life partner I don't want to imagine what life would have been without you. Your endless support and care provide me the energy to carry on and your honesty and kindness keep me grounded. I hope I am half as good as you in supporting your dreams, as you have been in supporting mine. I look forward to the journey ahead with you by my side.

Finally, I want to thank my huge family for not letting a day pass by without making me realize that I am loved regardless of what I do. I blame you all for spoiling me. I am so fortunate to have loving parents, grandparents, uncles and aunts, siblings and cousins, and nephews and nieces. Now that I have a Ph.D., I will try to act wisely, engage you in (slightly) meaningful debates, and be a better role model. For your affection, I am returning to India soon.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

SUMMARY

Motion gestures, detected through body-worn inertial sensors, are an expressive, fast to access input technique, which is ubiquitously supported by mobile and wearable devices. Recent work on gesture authoring tools has shown that interaction designers can create and evaluate gesture recognizers in stationary and controlled environments. However, we still lack a generalized understanding of their design process and *how to enable in situ and context-based motion gesture design.*

This dissertation advances our understanding of these problems in two ways. First, by characterizing the factors impacting a gesture designer's process, as well as their gesture designs and tools. Second, by demonstrating rapid motion gesture design in a variety of new contexts. Specifically, this dissertation presents: (1) a novel triadic framework that enhances our understanding of the motion gestures, their designers, and the factors influencing design of authoring tools; (2) the first ever explorations of *in situ* and *context-based* prototyping of motion gestures through development of two generations of a smartphone-based tool, *Mogeste*, followed by *Comoge*; and (3) a description of the challenges and advantages of designing motion gestures *in situ*, based on the first user study with both *professional* as well as student interaction designers.

CHAPTER 1

INTRODUCTION

Mobile and wearable devices accompany our multisensory bodies in all rich life experiences that occur in 3 dimensions, yet our interactions with them are mostly limited to 2-dimensional touchscreen interactions and occasional voice commands. As shown in Figure 1, the previous two generations of interaction styles—the Windows, Icons, Menu, Pointer (WIMP) style followed by the multi-touch and speech input style—have reduced the human body to a poking and speaking device with additional capabilities to see and listen. However, with the current generation of wearable and handheld devices, we have an opportunity to unlock sensor-enabled whole-body interactions (also referred to as embodied interactions [37, 41, 73]) such as motion-based gestures.

Though motion-based gestures can be sensed (e.g., camera [126], acoustic [49]) or interpreted (e.g., on a 2D surface like touchscreen [133], or in 3 dimensions) in several ways, for the purpose of this thesis, I define it as follows:

> *Motion gestures are intentional, voluntary 3D movements of any body part(s) detected through body-worn inertial sensors (accelerometers and gyroscopes).*

Motion gestures are different from WIMP and touchscreen input, in that they are more naturally eyes-free. They are also potentially very expressive and quick to access as an input technique. In general, free-form motion gestures "are likely to help users think and communicate" [73]. They are accurately characterized by 4-dimensional space consisting of 3 spatial dimensions plus a temporal dimension. Moreover, not only are they performed out in the world, but they also can involve the entire body. These advantages also present design and implementation challenges, such as identifying which types of motions that can be reliably performed by a body part, issues of social acceptability and difficulty in implementation (accurate recognition), due to factors such as mobility.

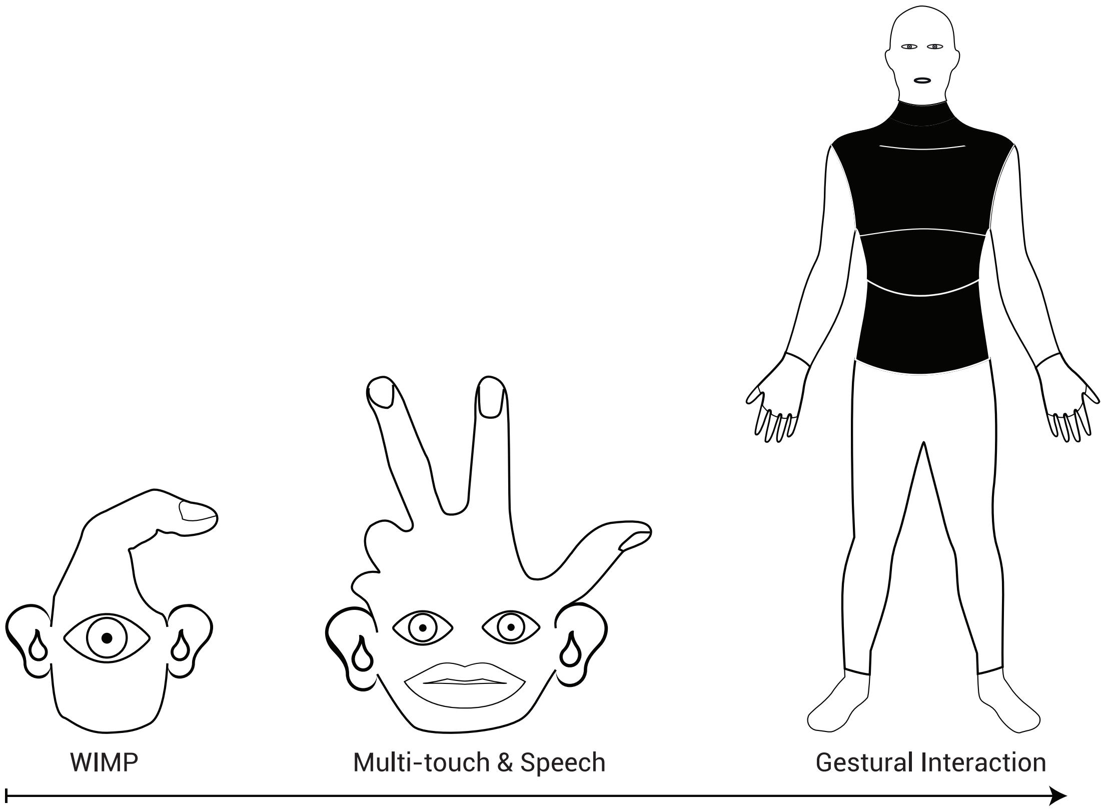


Figure 1: Evolution of interaction styles (left to right). We are progressively moving towards sensor-based interactions that enable whole-body experiences such as gestural interfaces. The figure for WIMP is adapted from O'Sullivan & Igoe's *Physical Computing* book [105].

Interaction designers can play a key role in ideating, prototyping, and evaluating usable, and appropriate gestures for end-users of mobile and wearable devices. But their lack of understanding of the inertial sensors and lack of expertise in programming and pattern recognition, both of which are deemed critical in the creation of recognizers for such sensor-based interactions, is a hindrance. Numerous research efforts, including the development of visual authoring tools (e.g., MAGIC [7] and Exemplar [52]), have tried to address these concerns using primarily a *programming by demonstration* [30] approach, where a designer teaches a computer a gesture by performing it several times. But the reality remains that it takes professional designers months to prototype and refine a single gesture, which may not have even been tested in a naturalistic context.

In this thesis, I argue there is a benefit to allowing designers to develop motion gestures *in situ*—as opposed to designing in the lab and later testing in the field. Abowd [1] supports my argument by suggesting that while programming new interactions such as motion gestures, designers should be "using tools that exactly match the characteristics of the end user experience." Until that happens, a developer/designer will be only able to make informed assumptions about end-use situations, but will not be able to simulate the exact social and cultural cues in a controlled environment. Thus, my hypothesis is that, when motion gestures are designed out of context, designers miss important contextual cues that can cause well-designed gestural interfaces and best-performing gesture recognizers to fail during real world use.

Researchers in mobile and ubiquitous computing have long identified the importance of contextual information sensed from the physical and computational environment in mobile interaction design [2], but there is little known about whether any contextual factors have an effect on a gesture's design and a designer's process. If so, in what ways? LaViola [84] proposes the idea of using contextual information gathered from a virtual environment for improving gesture recognition accuracy for *Kinect*-based (depth sensing camera that tracks the motion of the skeletal joints) gestures during game play. To my knowledge, no prior research describes tools that leverage context during the process of motion gesture design, the topic of this thesis research.

1.1 Thesis Statement & Research Questions

Based on the identified opportunities, my thesis statement is as follows:

> **A mobile motion gesture design tool that supports in situ, context-based prototyping will enable rapid creation of gestures for wearable/handheld devices and reflection on their usability, appropriateness, and recognizability.**

I further break down each aspect of this statement into a research question, which will drive an independent investigation to provide justification to my claim. In this thesis, I consider the following research questions.

RQ1. *What design guidelines and process do designers currently follow while working on a novel motion-based gestural interface?* (Chapter 3 & 4)

RQ2. *Can a mobile motion gesture design tool support rapid in situ prototyping of motion gestures with commodity wearable devices?* (Chapter 5)

RQ3. *What impact does context have on a gesture's design and a designer's gesture prototyping process?* (Chapter 6)

In Table 1, I summarize the research activities I have conducted along with the evaluation methods that I employ to answer each question.

Table 1: Research plan consisting of research questions, completed work, and evaluation methods.

Research Question	Outline of Research Activities	Methods
RQ1. What design guidelines and process do designers currently follow while working on a novel motion-based gestural interface? (Chapters 3 & 4)	Developing an understanding of motion gesture design from different perspectives and identifying important dimensions for tool design.	Interviews with professional designers and developers as well as novice designers. Qualitative analysis with open coding.
RQ2. Can a mobile motion gesture design tool support rapid in situ prototyping of motion gestures with commodity wearable devices? (Chapter 5)	Evaluation of the first iteration of a smartphone-based tool that allows the creation of motion-based gesture recognizers through programming by demonstration.	A user study with novice designers to gather qualitative experiential data and usability issues.
RQ3. What impact does context have on a gesture's design and a designer's gesture prototyping process? (Chapter 6)	Identification of contextual factors relevant to the motion gesture design process and building support for their capture and access in situ within next iteration of the tool. An evaluation with professional and student designers including free-form outdoor use in multiple mobility conditions.	A mixed-method user study with professional and student designers to measure the effect of contextual information on a gesture's design and the process. Quantitative of logs, app data, and questionnaire responses. Qualitative analysis with an open coding of the interview transcripts.

1.2 Summary of Contributions

In this thesis, I make the following contributions to the space of motion gesture research.

1. A triadic framework (see Figure 2) consisting of gestures, designers, and tools that provide a richer understanding of the types of gestures, considerations for and the

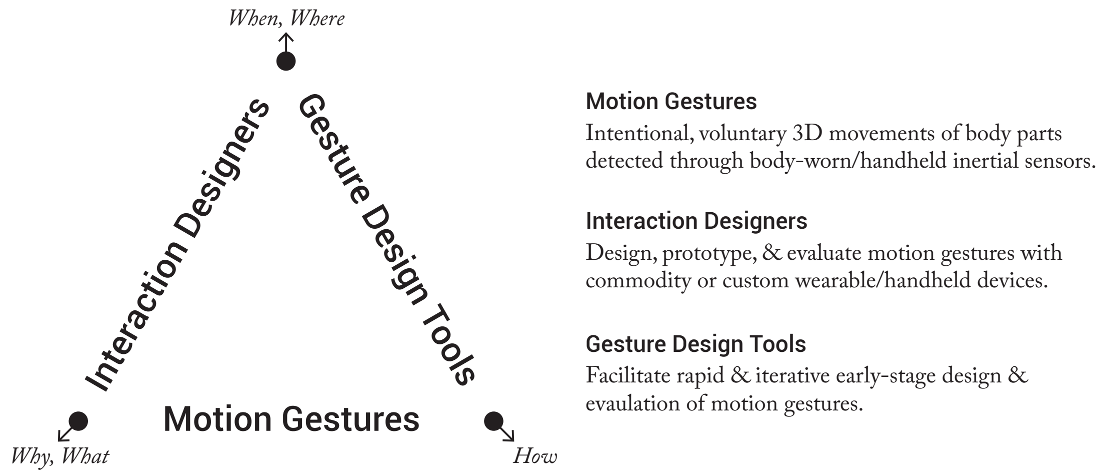


Figure 2: A triadic framework that visualizes the key entities in the gesture design process along its edges and opportunities that arise as a result of their interactions on the vertices. Each of the three entities are also defined in the right half.

process of gesture design, as well as factors that affect the design of gesture design tools.

2. The first ever explorations of *in situ* and *context-based* prototyping of motion gestures through the development of two generations of a smartphone-based tool, *Mogeste*, followed by *Comoge*.

3. A description of the challenges and advantages of designing motion gestures *in situ*, based on the first user study with both *professional* as well as student interaction designers.

1.3 Acknowledgment of Collaborators

This manuscript presents the work I have led, but I couldn't have done it alone. I have been fortunate to have numerous collaborators including my advisors, master and undergraduate students, and external collaborators who have significantly contributed to the work presented here. Their contributions are acknowledged through authorship in publications resulting from our work. However, henceforth, in this document, I only use the first person singular.

1.4 Thesis Organization

Chapter 2, presents a survey of relevant literature on several topics including techniques for sensing motion gestures, and tools for prototyping gestures, developing context-aware applications, early-stage interface design, and gestural interface design. Chapter 3 presents a formative qualitative study that contributes a deeper understanding of the motion gesture design process from the joint perspective of novices and experts. Chapter 4 describes a design space of gesture design tools and places related work within it. With a new understanding of the design process, Chapter 5 details motivation, design, implementation, and summative evaluation of an early version of a smartphone-based motion-gesture prototyping tool, called *Mogeste*, for interaction designers. Based on the insights gathered while developing and testing the tool, Chapter 6, presents the next iteration of the smartphone tool, called *Comoge*, that enables both *in situ* prototyping as well as capture and access of contextual information. Finally, Chapter 7 concludes the thesis.

CHAPTER 2

RELATED WORK

Gestural input is inherently quick and expressive since a single motion can indicate the operation, the operand, and additional parameters [18]. For instance, a simple turn of the wrist (gesture) quickly towards the eyes (parameters), expresses the wearer's intent to check the time (operation). As a result, the screen of a smart watch (operand) turns on to reveal the time. Motion gestures are particularly useful in situations where visual attention is limited [101], or when brief interactions with wearables are needed [7]. Moreover, novel applications of motion gestures include input to immersive systems such as virtual reality headsets (e.g., Oculus Rift) or gaming consoles (e.g., Nintendo Wii) and cross-device interactions such as bumping objects together to transfer information [54]. In general, free-form motion gestures "are likely to help users think and communicate" [73].

2.1 Sensing of Motion Gestures

Within research literature, a large variety of sensing techniques for gestural interaction have been explored. Most common gestural interfaces can be characterized as touch- [133], pen- [56], tangible- [119], or motion-based [124]. Combinations of these modalities have also been explored (e.g., pen with touch [61], touch with motion [59], touch plus in-air [25, 58]). Although researchers have demonstrated use of electric field [27], electromagnetic field [28], capacitive [111], and acoustic [49] sensing for detecting hand gestures, cameras (cf. [96, 124, 126]) and inertial sensors are most common.

Cameras or more generally, vision-based sensing techniques are probably the oldest. Lately, Kinect and other depth sensing approaches have made whole body interaction popular. However, this modality raises privacy concerns and is also computationally intensive, thus, making it impractical for mobile and wearable devices. More recently, Project Tango, an experimental tablet with 3d depth sensing capabilities has shown promise in making mobile vision-based gestural interaction possible.

Inertial sensing is more practical and affordable computationally and monetarily. Inertial sensors (e.g., accelerometers, gyroscopes) are ubiquitous in handheld and wearable devices today. Their lightweight and small form factor allows them to be strapped to a body part to provide direct sensing of its motion. Although inertial sensors are commonly employed for activity recognition [16], their use for gestural interaction on commodity devices is limited to specialized functions (e.g., checking the time by rotating the wrist) or gaming. We have begun to see inertial sensing leveraged for motion gesture input with a growing set of gestures, available on Android smartwatches. In the research literature, they have been utilized for detecting taps on and around a device [138], coarse motions such as writing in the air [4], or motion states [80] such as walking, running, or driving to infer interaction needs. With Mogeste, I explicitly support the use of inertial sensing for designing motion gestures.

2.2 User Interface (UI) Programming Approaches

UI development tools allow designers to specify complex interaction behaviors in two distinct ways: programming *by demonstration* and *by declaration*. In programming *by demonstration* (PBD) approach the designer instructs the system to "watch what I do"[30]. In other words, the tool learns a behavior from representative examples provided by a designer. For example, with Exemplar designers authored sensor-based interactions such as head tilt using PBD [52]. Here the onus is on the tool to infer generalizations from recorded user actions. Rubine used this insight to allow programming of touchscreen gestures [114]. More recently, Gesture Coder demonstrated application of PBD to multi-touch interactions with additional support for generating source code that developers can use to handle gesture events [93]. Both MAGIC [6] and Exemplar [52] extended this idea to the realm of 3-dimensional motion gestures.

In the past, Myers demonstrated use of PBD for specifying graphical user interfaces through Marquise [100]. Further, Lau conceptualizes PBD as a machine learning problem and highlights the distinction between its two implicit notions: observing user's actions versus observing changes in the application state during demonstration [83]. Mogeste ascribes to former notion of PBD by creating motion gesture recognizers from a few samples of motion recorded from inertial sensors on commodity devices. Furthermore, it allows a designer to edit, and test invocation of these recognizers. One challenge with PBD is that it suffices for simple behaviors, but is not sufficient by itself for learning complex behaviors such as multi-device interactions.

In contrast to PBD, programming *by declaration* allows a designer to state "what should happen" [99] using a high-level specification language. Note the emphasis on the description of the behavior as opposed to the underlying procedural specification that concerns itself with "how to make it happen." Recently, declaration-based programming approach has been utilized for specifying multi-touch interactions [94, 72, 71, 120]. Midas provides a declarative rule-based model for programming new and composite gestures [120]. Proton, on the other hand, uses regular expressions of primitive finger movements to program multi-touch gestures [72, 71]. Furthermore, Proton++ incorporates attributes such as trajectory and touch shape into the regular expression for additional functionality including user identification in a multi-user scenario. Although declarative languages can handle complex gestures, the specification itself can become difficult to comprehend.

Finally, Gesture Studio provides an example of combining these two complementary approaches for programming basic and compound gestures respectively [94]. Lu and Li employ a video-editing metaphor for providing a sequential and parallel composition of gestures.

2.3 Tools for Prototyping Gestures

Prior literature has adequately described the design space for touch surface gestures (e.g. [133]), but similar work on motion gestures has been limited. Ruiz et al. [116] proposed a taxonomy of user-defined gestures for common tasks on mobile phones. However, unlike surface gestures, the three-dimensionality of motion gestures makes the task of authoring them non-trivial.

Historically, authoring and prototyping have been mostly tackled by machine learning experts or advanced developers. Recently, several attempts have been made to streamline the motion gesture implementation pipeline for non-experts. For such interventions, programming by demonstration [30] has been the common approach used to implement tools for gestural interactions performed by non-experts. The Gesture Recognition Toolkit [43] is an open-source C++ library designed to make real-time machine learning and gesture recognition more accessible and easy to use for non-experts. Exemplar introduced the design-test-analyze workflow for authoring of sensor-based interactions [52]. Through a direct manipulation interface, it enabled novices to rapidly explore gestural interaction with a variety of sensors. MAGIC [6], a desktop tool for creation, testing, and evaluation of motion-based gestures with a wrist-mounted 3-axis accelerometer, built on Exemplar in two ways. First, it exposed the machine learning (ML) measures of inter- and intra-class comparison scores and confusion matrices in a graphical representation. Second, it introduced a comparison of intended gestures against a repository of everyday hand motions in the analysis stage. Wekinator [40] is another tool, which uses the Weka library [50], built for musicians and creative coders. It enables on-the-fly learning using various input sources, such as gestures and joysticks, to generate output fed to an audio synthesis instrument.

My hypothesis is that mobile tools can bring usage and development environments closer. Kim et al.'s *M.Gesture* system [68] provides the first mobile interface for accelerometer-based gesture authoring with multiple mobile devices. It lets users specify a gesture trajectory and planes (or hurdles as introduced by *EventHurdle* [69]) it must cross, following a mass-spring analogy. My own work, *Mogeste* [106], also a smartphone-based gesture authoring tool but using both accelerometers and gyroscopes, introduces the idea of *in situ* motion gesture design. Amini and Li's *CrowdLearner* [3] proposes a novel framework for developers which leverages crowdsourcing to rapidly generate mobile recognizers for specific tasks and gestures, minimizing the overhead of recruiting participants and performing data collection in person. Although these explorations provide means to record gestures in naturalistic contexts, they do not record any contextual information while in situ. Unfortunately, if the context of use is not considered during the design of gestures, the burden of which gestures to use in an interface falls on application developers. This observation is a key motivator for the development of Comoge as a prototyping tool for designers of next generation gestural interactions.

2.4 Contextual Interaction Design

Researchers in mobile and ubiquitous computing have long identified the importance of contextual information sensed from the physical and computational environment in mobile interaction design [2]. Dey, Abowd, and Salber [33] provided a definition of context that includes "location, identity, and state of people, groups, and computational and physical objects". They also introduced a toolkit that made the development of applications that use context easier for non-experts [117]. Selker and Burleson [123] initiated a discussion on how contextual awareness, embodied in models of the task, user, and system, can be leveraged to design artifacts and services that provide good user experiences. Dey et al.'s *a CAPpella* [34], on the other hand, exemplifies *programming by demonstration* [30] as a viable approach for development of complex context-aware behaviors by non-expert end-users. It allows in situ recording and annotation of representative behavior without having to write code. Comoge takes a similar approach, allowing in situ recording and annotation of motion gesture designs.

Sensing is a key enabler of contextual computing. HCI researchers have explored novel sensor-based interaction techniques that leverage context. For instance, in an early demonstration, Hinckley et al. [58] envisioned use of touch, tilt, and proximity sensing for adapting content display and increasing input vocabulary of mobile devices. Recently, Goel et al. demonstrated the use of existing sensors (e.g., accelerometer) on smartphones and contextual factors, such as handedness [45] and activity context [44], to improve and augment touchscreen interactions. Similarly, Comoge facilitates the use of multi-factor contextual information for creation of usable, and appropriate gestures.

LaViola [84] proposes the idea of using contextual information gathered from a virtual environment for improving gesture recognition accuracy for *Kinect*-based (depth sensing camera that tracks the motion of the skeletal joints) gestures during game play. Lu and Soo [92] suggest the use of temporal, spatial, social, emotional, and behavioral context while creating an interactive storytelling game. To my knowledge, no prior research describes tools that leverage context during the process of gesture design, instead, they only utilize context for either adapting and augmenting existing interactions, or selection and recognition of gestures from a predefined database of motions.

2.5 Early-Stage Interface Design Tools

Mogeste also draws on the long history of research on user interface design tools. Here, I describe a few illustrative examples. The very first example of an interface design tool was SketchPad, constraints based system developed by Ivan Sutherland [127]. In the PC era, Myers and colleagues conducted pioneering work in techniques and tools for graphical user interface development by experts and novices alike [99, 98]. For instance, SILK allowed designers to rapidly sketch interface elements which were then converted to their functional equivalents [81, 82]. Landay and colleagues, among others, carried forward this work into the ubiquitous computing era [74, 86, 91, 118].

With Quill, they enabled designers to program pen-based gestures by demonstration, while the tool simultaneously analyzed incoming gestures for perceptual and recognizer similarity [91]. Next, SUEDE allowed designers of speech user interfaces to rapidly prototype their designs using a Wizard-of-Oz technique [74]. This work also contributed a design/test/analysis methodology typically followed by designers of recognition-based interfaces. In addition to MAGIC, and Exemplar, Mogeste also supports this workflow. Subsequently, with ActivityDesigner [86], they enabled "rapid creation of (ubicomp application) prototypes based on modeled activities by allowing designers to specify stream-based interaction behaviors easily via direct manipulation and an activity query language." More recently, they investigated muscle-computer interfaces using electromyography sensing for enabling always-available low bandwidth input [118]. During this time, other application themes emerged which led to new prototyping tools.

One long-standing application theme is of context-aware applications [2]. Dey, Abowd, and colleagues have done pioneering work in this domain [32, 33, 34, 35, 87, 79, 117]. Some of their early work defined different types of context such as who, what, when, and where as a proxy for the user's intent (why) [33]. They also contributed first context-aware toolkit that enabled development of context-enabled applications [117]. Their toolkit provided context widgets, analogous to GUI widgets, that encapsulated sensing, and interface callbacks. They further demonstrated several applications developed with their toolkit. Next, a CAPpella supported PBD paradigm for developing context-aware applications [34]. Subsequently, iCAP provided visual programming support for setting if-then rules [35]. More recently, they have contributed toolkit for extending intelligibility of context-enabled behavior by end-users [87]. Lastly, they released AWARE framework, an open source tool for collecting context information from mobile devices in an application agnostic manner[1]. I leverage some of this work for the development of context-aware motion gesture prototyping, called Comoge (Chapter 6).

Finally, I highlight some of the commercial tools that have fulfilled the need for early-stage mobile and web application prototyping. Pop[2], Pixate[3], and Invision[4] provide extensive support for wireframing mobile and web applications through their mobile and desktop tool offerings. All of them allow designers to create or upload screen mockups, create hotspots (active regions), and link screens together through touchscreen interactions. The same interactions can then be used during run-time for triggering transitions between or within screens.

Despite such a rich history of UI tool development, there are some gaps that I have identified. First, is a lack of tools for end-to-end wearable interaction design. Although some of the tools for mobile could be appropriated for this purpose– as done with popular developer tools such as Android Studio and XCode– there exists an opportunity for defining new tools for enabling designers to create novel experiences for this emerging computing paradigm.

[1] `http://www.awareframework.com/`
[2] `https://marvelapp.com/pop/`
[3] `http://www.pixate.com/`
[4] `https://www.invisionapp.com/`

2.6 Evaluation of UI Tools Research

Here, I will enumerate the metrics for success and value of UI systems research as provided by Olsen [104], Myers et al. [98], and Hudson and Mankoff [63]. Olsen suggests that evaluation of UI systems should focus on importance, reducing solution viscosity, novelty, generality, enabling scale, and ability to empower new design participants [104]. Referring to the Situations, Tasks, Users (STU) framework, Olsen explains that a system should be important to different types of users, higher frequency tasks for a large population of a single set of users, or several common or underserved rare situations for a smaller set of users. The principle of novelty exclaims use for an unsolved problem. The generality principle proposes utility for solving a diverse range of problems. The principle of reducing solution viscosity refers to the ability to develop good solutions faster. Enabling scale requires the solution to think about a large user base or a bigger problem. Lastly, a system should be inclusive of a new population.

Myers et al. highlighted the principles of the lower threshold, higher ceiling, and provides the least resistance to good solutions (similar to reduce solution viscosity) [98]. A lower threshold means reducing the entry barrier thus providing faster learnability. A higher ceiling means, once learned a system should allow a user to do more complex and diverse things. The higher the ceiling the more utility one gets out of the tool.The principle of least resistance means supporting developers in doing rights things and reducing errors.

Hudson and Mankoff suggest lower threshold, higher ceiling, breadth of coverage, increased automation, and good abstractions or extensibility [63]. According to them, after proof-of-concept implementation has been provided an evaluation of a tool should focus on demonstrating "simplicity of creation, power or complexity, or the variety". For example, Phidgets [48], an early toolkit for building physical user interfaces, claims easier development (simplicity), and demonstrates domain-agnostic use (power and complexity) in a variety of projects by students. Furthermore, good abstraction can be validated through descriptions and comparison to related work. Lastly, usability for developers can be validated through releasing the source to the developer community and letting uncontrolled use.

Notably, Olsen [104], Myers et al. [98], Hudson and Mankoff [63], and Greenberg and

Buxton [47] suggest holding off on the usability testing until the work has matured sufficiently otherwise there is a risk of hampering its progress or even killing it in its infancy. I believe most of the motion-based gestural interaction design work falls into this category.

2.7 Summary

In this chapter, I have presented a survey of prior research related to this thesis. I covered different techniques for motion gesture sensing followed by a discussion on common UI programming approaches. Next, I highlighted developments in prototyping tools for gestures and early-stage interface designs. Finally, I discussed metrics for evaluating UI systems research that I have applied to my work. During this literature review, I identified several gaps including a limited understanding of a designer's process for prototyping motion gestures, lack of tools to support in situ development, and limited support for end-to-end wearable interaction design. Rest of the thesis describes my attempts to fill this gap through an analysis of qualitative data from designers (Chapter 3), extraction of a design space to inform the development of gesture design tools (Chapter 4) and development of tools (Chapters 5-6).

CHAPTER 3

UNDERSTANDING MOTION GESTURE DESIGN PROCESS

When I started this work with an extensive review of prior research, I found several examples of tools and techniques developed for the purpose of creation of gesture recognizers by non-technical experts motivated by their lack of expertise in programming and pattern recognition. Surprisingly, there was very little coverage of the process of gesture design from a designer's perspective. My hypothesis was that without a thorough understanding of the existing process and challenges faced, a gesture design tool will fall short of expectations. This observation led to my first research question, **RQ1. What design guidelines and process do designers currently follow while working on a novel motion-based gestural interface?** This question is investigated in two parts: 1) the current chapter presents findings from interviews conducted with both expert and novice gesture designers; and 2) the next chapter collates these findings with my own experience building gesture recognizers to provide a design space of motion gesture design tools.

3.1 Introduction

Until now the development of tools for enabling motion gesture prototyping by designers was based on three insights: 1) designers lack programming and pattern recognition expertise, 2) programming by demonstration makes it easy for them to train recognizers, and 3) designers follow a create-test-analyze workflow. Although I agree with this synopsis, I notice a few gaps in the understanding of a designer's process. First, what is the difference in a novice's process as compared to an expert's process?. Second, since designers are inherently resourceful, how does a designer go about conceptualizing and communicating a gesture without access to advanced gesture prototyping tools?. Third, when a designer collaborates with a developer(s), how does her process change?.

Motivated by the questions posed above, my goal in this chapter is to provide a unified and deeper understanding of the motion gesture design process as experienced by a particular audience—interaction designers. To address this goal, I conducted and analyzed formative interviews with experts and novices to identify their challenges and needs. A rich description of their processes and considerations while designing gestures emerged from my data. Additionally, I identified specific dimensions that can be used to describe a motion gesture prototyping tool and formulate a design space. I use the design space to position prior work and provide opportunities for future tool development.

In this chapter, I contribute an understanding of the motion gesture design process for novices and experts through formative interviews.

3.2 Formative Interviews with Experts & Novices

I collected rich, qualitative data from both experts and novices to uncover critical aspects of the motion gesture design process and unmet needs. Two separate studies are described below. I discuss insights in the following section.

3.2.1 Participants

I interviewed 13 participants in total (6 female). My first five participants were professionals (PP1-6) from a large technology corporation located in the San Francisco Bay Area, California. All of the participants had experience creating motion gestures for wearable devices. My participants consisted of software engineers and product managers with degrees in machine learning, interaction designers, and former HCI researchers. Henceforth I will refer to them as expert designers (PP1, PP4), developers (PP2, PP3, PP5), and researchers (PP5). The remaining seven participants were students (SP1-7) in their second year of a Master's in Human-Computer Interaction degree at my institution. These students had no prior experience designing gestures, so I refer to them as novices.

3.2.2 Protocol

In the first phase, I conducted open-ended interviews with the experts. During these interviews, I sought to understand their current process and challenges and to note their tools, setup, and unmet needs. My interviews with the professionals provided useful insights into an experts' process. However, my understanding of the novices' (designers who haven't designed gestures before) process was shallow.

Thus, I conducted semi-structured interviews with seven novices. To seed the discussion, participants were introduced to motion gestures and asked to design 3 alternative gestures for two tasks on a mobile phone: *placing a phone call*, and *ending a phone call*. They were provided a phone to brainstorm with, and a worksheet to describe their designs on. I used a think-aloud protocol to elicit their thought process. During the interviews that followed, I focused on understanding their design rationale and how would they implement and evaluate the designs.

3.2.3 Analysis

I audio recorded all interviews and also video recorded interviews with novices. Both a colleague and I went through all recordings and took notes, and then transcribed them. Based on the notes and open coding of two out of five expert interviews, I developed an initial codebook (see Table 2). Subsequently, the we coded the remaining ten transcriptions together using the same codebook and reached agreement collaboratively in real time. When disagreements occurred they were discussed and resolved.

My initial code book had seven themes. After coding the remaining three professional interviews I added two more. While transcribing the novices' interviews, codes were only refined and broadened, indicating my code book was adequate in describing the remaining interviews.

My expert participants were a heterogeneous group and each interview with them led to newer insights, suggesting the need for further interviews. In contrast, my novice participants were a homogeneous group; I hit saturation with them after the fourth interview.

Table 2: Emergent themes. Three main considerations (categories) while designing gestural interaction are: 1) Understanding, describing, defining, & identifying gestures, 2) Implementation of gesture designs, and 3) Consideration of stakeholders including designers (primary), developers, and end-users. Several themes that emerged from qualitative analysis of interview transcripts naturally fall under these top-level categories.

Category(C)	*Theme(T)*	*Description*	*Example*
Gestures	*Meaning making*	Designers associate some meaning with an abstract idea of a gesture.	*I will think of a metaphor like watching time.*
	Considerations for gesture design	Designers identify constraints that both inform and scope the exploration space of candidate gestures.	*I might find a gesture easy to perform and detect but not socially acceptable.*
	Communicating a gesture	Designers choose some methods for communicating to others or reminding herself a gesture.	*I will show them a video or sketches with description.*
	User research	Designer does research to identify appropriate gestures.	*I will observe what gestures do people normally perform.*
Implementation	*Rapid prototyping*	Designers rapidly prototype a gesture with available means.	*My goal is "idea to prototype in 3 hrs" [PP4]*
	Improving accuracy	Designers can help improve the accuracy of gesture recognition.	*I will shake it twice at least because once is not enough.*
	User evaluation	Designers evaluate their gesture set along with several dimensions.	*I want to know is the gesture doable and not awkward.*
Stakeholders	*Team communication & collaboration*	Designers collaborate with others on gesture design team, when they lack necessary skills.	*I will prototype ideas myself if developers are not convinced. [PP1]*
	Empowerment	Designers want to decrease dependence on others by using appropriate tools.	*I might not know where to start from.*

3.3 Themes for the Design Process

I collate all formative insights gathered from the two studies described above into frequently occurring themes and a few higher level categories (see Table 2). Note that in doing so, my primary focus is on interaction designers. However, in some places, I will use developers' and end-users' perspectives to illustrate a point.

Similar to related work, I found evidence for the use of the create-test-analyze workflow [74] and programming-by-demonstration [52] concepts, but several other salient themes emerged (see Table 2). Upon coding for these low-level themes (T1-9), I identified a natural grouping with three high-level categories (C1-3).

3.3.1 C1: Understanding, Describing, Defining, & Identifying Gestures

This category covers specifics of gestures, focused on how a gesture is semantically understood, communicated to others, defined within constraints, and identified appropriately.

3.3.1.1 T1: Meaning making

What does a gesture mean? When thinking about a new gesture, a designer's instinct is to find everyday words, actions, and metaphors that even remotely relate to the task for which gestures are being considered.

> *"So to place the call ... the most intuitive way I am thinking like...(brings phone close to the ear and repeats the gesture multiple times) because of this how I hold my phone to call somebody."* [SP6]

Finding colloquial phrases to explain a gesture allows designers to instantly develop a common understanding with others and also helps in recalling how the gesture was performed. For example, verbs such as *wave*, and *shake* are commonly used to describe gestures. When designers are thinking about gestures for a technology that is not well understood, designers tend to define concepts to the bootstrap development of a shared understanding among the design team. For example, [PP4] defined the concept of "world-fixed" to explain how a head-mounted computer would feel like.

Designers strive to find something that is “intuitive”, “natural”, or “normal”. Failing at that, they would settle for something analogous to natural, meaning some slight variation. Unless designing gestures for a specialized domain, they rely heavily on their personal experience and search for *meaningful* gestures. While doing this, they try to avoid weird and difficult-to-perform gestures.

When at a loss as to what gestures to design, they seek inspiration from a known set of gestures, if one exists. They would look at existing interactions, for example, on a touchscreen, in hopes of porting them over to the motion gesture domain. Since many of them are not natural gestures, they would use them only if they understand the underlying interaction model. For example, pinch-to-zoom suggests an interaction model which is easy to remember once understood [PP5].

A common strategy they apply when a natural pairing of tasks exists, such as placing and ending a phone call, is to utilize opposite gestures. Moreover, if the paired tasks can only occur sequentially, such as ending a call once it has been placed, they would utilize the same gesture for both tasks.

3.3.1.2 T2: Considerations for gesture design

What are good gesture candidates? The design of gestures is informed by criteria the designer deems important and by constraints determined by envisioned applications. While expert designers are concerned with defining interactions that are expected to work reasonably well for a diverse set of potential users, devices, and situations, my novice participants consider the context of use of a gesture as their primary consideration.

Constraints applied to motion gestures can be broadly grouped into the following categories: hardware, context, physical constraints, and characteristics of a gesture. First, within hardware, knowledge about which sensors to use, and what is the intended device form factor for given gestures are important decisions. An expert participant reflects that although interaction design and hardware specifications are supposed to be interdependent, often hardware specifications are fixed early in the development process, which significantly narrows the interaction design space.

Secondly, the space of gesture exploration can be naturally defined by social, cultural, and application contexts. Moreover, the context of an activity can be explained based on the mobility & dexterity. Mobility describes situations where the user is moving or not, for example standing versus walking. Dexterity refers to the user's ability to use their hands while they are hands-free or hands busy. Within this framework, a designer might focus only narrowly on the specifics of an application or broaden the scope to consider the entire user experience, allowing them to consider outlier cases. As an example, a participant describes:

> *"different scenario people are making phone calls like holding other stuff in your hand or walking very fast or in a very crowded subway"* [SP1]

In contrast, a designer might be working on a universal gesture that works across different technologies, people, cultures, and use cases.

Thirdly, physical characteristics and constraints of the human body might rule out certain gestures. For example, a person's "head doesn't translate" [PP4]. Additionally, differences between people cause inter-personal variations in how a gesture is performed. In fact, the same person might perform the same gesture slightly different today compared to how they performed it in the past. For these reasons, my expert participants aim to capture these variations and then define accuracy-based factors, such as age and gender. A participant also noted that variations in a user's emotional state may cause performance differences for the same gesture; for example, in cases of mental and physical fatigue. Therefore, a designer's goal is to find gestures that are comfortable by reducing the physical effort needed and minimizing variations due to fatigue.

Lastly, besides physical, contextual, and technical constraints, there are inherent dimensions along which two gestures can be compared. These include the axis of movement, start and end position, speed, angle, and duration. Additionally, some meta parameters such as fluidity and subtlety may be relevant for discrimination. However, subtle differences along these dimensions may not be noticeable by a person. Despite this, [PP1] preferred subtle gestures for personal use. As [PP3] begrudgingly noted, "things kind of get confusing when people can't differentiate the speeds at which you are supposed to move."

An obvious starting point for picking a reasonable gesture set is to define gestures inspired by natural actions and habits. However, a downside of this approach is that it may result in confusion between intentional and unintentional gestures (i.e., everyday actions). I describe strategies for improving recognizer accuracy in theme T6.

End-user control is another relevant dimension for gesture design. A designer may ask "should I give the user control or decide for them?" In the latter case, a designer may identify reliable, performable gestures for end-users. In the former case, she might consider providing defaults and allowing the end-user to create a personalized gesture set.

Finally, my participants pointed out that "gestures and speech co-occur" [SP4, PP4] for humans. What does that mean for gestural input? This finding suggests designers should be cognizant of the social acceptability of a gesture.

3.3.1.3 T3: Communicating a gesture

How are gestures communicated to and remembered by human beings? This theme relates to methods of communicating how a gesture is performed rather than its meaning. Articulation of a gesture's motion or trajectory is particularly important when:

- asking others to perform a gesture repetitively during a data collection exercise;
- seeking feedback from team members on several design alternatives [PP1];
- trying to recall a previously designed gesture [PP3]; and
- creating a tutorial for teaching at scale a gesture that's made available on a device [PP3].

The best way to communicate a gesture is to demonstrate it, preferably with an actual device. Remembering the "correct way" to repeat a designed gesture was a challenge across novices. This problem can be addressed by either using only gestures that have an unambiguously associated meaning or learning the underlying model which is also reinforced by consistency across platforms. For example, pinch-to-zoom on touchscreens is not a natural gesture but has now been adopted widely because it works the same way on all devices.

Unfortunately, either strategy does not work well for motion gestures. Firstly, if we use everyday actions as gestures, separating intentional gestures from unintentional ones becomes challenging. Secondly, since not many platforms provide support for motion gestures, there are no standard definitions for gestures. Additionally, differences in physical attributes such as arm length, or personal preferences such as subtlety, can cause significant variation in how a gesture is performed.

For those reasons alone, until motion gestures become ubiquitous and gesture recognizers become user-independent, designers will have to rely on other means to clearly communicate how a gesture is performed. Common methods that my participants collectively recited were textual descriptions, videos, pictures, icons with arrows, drawing/sketches, storyboards, and animated sequences. Expert designers have also worked with technical specification documents precisely describing a gesture and the operation it is supposed to trigger. A designer has to be cautious about using static representations because they inherently lack temporal information and can cause them to overlook other subtle differences between similar gestures. For example, a slow upward motion might be different than a fast upward motion.

When it came to teaching a gesture to a remote user, experts suggested the use of animated imagery and text, which matches my experience with tutorials on an Android smartwatch. Alternatively, a video of an expert's performance can be provided, similar to MAGIC [6].

Thus far, I have elaborated on multiple ways of conveying spatial and temporal information related to a gesture, but gesture designs often encode many more parameters such as cultural considerations. These factors can be very important for designing usable and distinguishable gestures. My next theme discusses these factors.

3.3.1.4 T4: User research

Finally, how are good gesture candidates identified? This theme relates to the stage where designers try to understand user's actions, behavior, and contexts. A common theme appeared across both novices and experts that they would conduct a user study. For example, capturing video in a public space to observe common gestures or asking the participants to perform the gestures to seek their subjective feeling or determine variations based on gender or age. Some participants were also critical about understanding why should motion gestures replace touch and when would motion gestures become relevant. Though most participants agreed on the utility of the user research, one expert participant argued to derive insights from personal use and experience, as designers are people themselves and thus understand "people" to some extent. He suggested using user research as an evaluation tool rather than an exploration tool to inform the gesture design process. There was a general consensus among all participants that to design gestures, understanding of user's social context, commonly performed gestures, and usage patterns would better inform the gesture design process.

3.3.2 C2: Implementing Gesture Designs

After a few good gesture candidates have been identified, the designer typically tries to rapidly prototype each one of them with available means so that the gesture can be *experienced*. Later, a subset of these candidates is put through technical evaluation (with help from developers) and a user evaluation process. The challenges that a designer is addressing here are discussed in the following themes.

3.3.2.1 T5: Rapid prototyping

What does a designer need to do to prototype a gesture set? Prototypes "embody design ideas or specifications, render them concrete and, in doing so, inform the designer's thinking" [73]. Furthermore, prototyping leads to unforeseen discoveries, makes communication with others easy, and enables testing of ideas. Four participants underscored this importance:

> *"The only idea that's worth discussing is the one that you can experience"* [PP4]

> *"Gesture design is not real until it is implemented"* [SP3]

Rapid prototyping, as the term suggests, is about building a prototype within a short time frame, in this case, a few hours. This iterative process facilitates consideration of more ideas in a fixed duration, thus increasing chances of finding a better idea faster. As one of the expert designers noted, her typical process is to go from an "idea to prototype in 3 hrs." However, the pace is dependent on skills and available tools.

As a case in point, for prototyping motion gestures a designer has to be skilled in pattern recognition and also have tools that support a create/test/analyze workflow. Though some designers are familiar with coding, pattern recognition is an advanced topic that even experienced programmers may not understand well. One expert designer had previously used a non-commercial tool that encapsulated pattern recognition knowledge and allowed her to focus on an exploration of the design space. In the absence of such versatile tools, designers team up with developers, but their collaboration is not straightforward, a point I elaborate upon in a later theme.

Let's say designers have no choice but to prototype these gestures themselves. In such a case, they improvise, experiment with what's available, and, if nothing else, use a wizard-of-oz. However, if the situation demands a higher fidelity prototype, then designers first and foremost seek answers to preliminary questions such as *what sensors to use?*, and *how would a gesture's motion be detected?* Then they have to implement a method of detecting relevant motions. In such a scenario, development and design for gestures co-occur. Irrespective of the method of implementation, when designing gestures for wearable and hand-held devices, being able to test on multiple devices is critical, which makes the development effort more involved. Once again, a tool could significantly reduce this effort.

Besides implementation support, my participants also emphasized the ability to experience interactions within an application, rather than in isolation. For example, they want to be able to actually place a call when they perform a gesture. Similarly, when characterizing the influence of each consideration for gesture design discussed above, my participants wanted to record gestures in the context they are designing for. For example, an expert wanted to test smartwatch gestures in situations when a user had only one hand free.

3.3.2.2 T6: Improving accuracy

How can a designer make gestures more reliable? Some strategies my participants use for reducing false triggers or avoiding accidental triggers are: a) use of compound gestures similar to ”double lock” [PP1]; b) defining orthogonal gestures (e.g., vertical motion, horizontal motion); c) using repetitive gestures; d) varying the duration of a gesture; and e) making the gesture trajectory pronounced and distinct from everyday motion. Additionally, activation gestures such as DoubleFlip [115] could be employed to delimit gestural interaction.

Another interesting way to ensure high accuracy is to determine thresholds corresponding to the range of movement of a body part, such as the flexion of the wrist. By providing example gestures, the tool can learn an expected range and seek to minimize the presence of outliers to ignore them.

The last set of techniques relates to rigorous testing. First, common knowledge suggests testing with multiple devices and distinct users in a variety of situations would help a designer account for the variance in how a gesture is performed. Moreover, [PP2] suggested building “an exhaustive corpus of everyday behaviors” and the ability to highlight problematic situations, which could be further evaluated. Similarly, identifying a general set of gestures, including culture-specific examples, will help find gestures that are easily separable from existing gestures.

3.3.2.3 T7: User evaluation

What gestures are appropriate for users, and why? This theme relates to the evaluation of the performability and recognizability of gestures by the user and the tool, respectively.

Two major ways of evaluating any gesture are to perform a sanity check (i.e., testing the gesture oneself) and to conduct an evaluative study collecting data from a representative sample of people. In both cases, data is captured in various social conditions or situations with variable mobility and dexterity to capture outlier cases for the gestures. The data collected from the study or self-use can then be used to evaluate the gesture using different metrics, including reliability, universality, comfort, ease of use, etc. These metrics are inspired by the section on considerations for gesture design (T2).

Expert participants shared additional insight, as they have either been part of prototyping a gesture recognizer or have worked with a team doing so. According to them, it was challenging to recall how the subjects performed in these situations. Taking field notes was also critical as they risked otherwise losing important contextual information.

3.3.3 C3: Considering Stakeholders

The previous two categories focused on the conceptualization and implementation of gestures from a designer's viewpoint only, but designers seldom work alone, particularly when they lack the much-needed expertise in pattern recognition. Consequently, instead of looking at a designer in isolation I situated a designer within a team of people who ideally have complementary skills. However, this means that a designer no longer controls the whole process and is influenced by other stakeholders, primarily developers. Two important questions that arise are addressed next.

3.3.3.1 T8: Team communication & collaboration

How can a designer collaborate effectively with others? This theme reflects on the communication and collaboration that takes place between developers and designers when they work in a team. This theme sheds some light on their interactions, expectations, and friction points.

Experts and novices both shared similar perspectives on their roles and the extent of team collaboration during a project's lifecycle. While experts spoke from their experience working in an industry setting, novices reflected based on their experience working on various academic projects and understanding of the expertise needed for the gesture design–implementation process.

My conversation with the experts revealed that any project is either initiated by the design team or the development team. If the design team initiates the project, developers are involved early on in the process with the designer seeking the developers' expertise (computer and hardware engineers). A designer goes to a developer with her designs, they implement a case, designers raise concerns over the prototype, and iterate with the developer. There are numerous back and forth conversations which may lead to friction between the designer and the developer, typically because each one uses different criteria to evaluate the designs. A developer may decode the designer's concerns in terms of machine learning parameters, whereas designers are not typically familiar with machine learning and discussing its challenges. Table 3 hints at these differences. This situation leads to considerable information disparity between developers and designers.

In some cases, designers need to provide justification for their design alternatives before a developer moves ahead with implementation. Developers can raise concerns about the design implementation by explaining constraints. Based on the developers' feedback, designers either move on to the next design alternative or revisit their assumptions as a sanity check to gauge realistic expectations from the developer.

A developer-initiated project will focus on prototyping a primitive version of the gesture recognizer and application before bringing other teams (e.g., design) into the project. At that point, they seek specific feedback on social acceptability, ergonomics or other metrics (covered in theme T2 above on considerations for a gesture).

Novices talked about a collaboration model between designers and developers. Novices most closely identified themselves in the role of researchers or interaction designers. Novices also reached a consensus that they would prefer to work in a team alongside developers in any gesture design project. A developers' understanding of activity recognition would facilitate them implementing the designed gestures.

Across all novice and expert participants (with a design background), I observed designers' lack of expertise to quickly prototype their envisioned gestures and evaluate them without help from a developer team member.

3.3.3.2 T9: Empowerment

What are the challenges for a designer trying to work alone? The design and development of motion gestures requires a team effort from different stakeholders (see Table 4). On one end we have *end-users*, on another we have *developers*, and finally, we have *designers* who act as mediators between the two.

Table 3: Most frequently mentioned gesture design and evaluation criteria by designer and developer.

Gesture Design & Evaluation Criteria

Designer terms	*Developer terms*
social/application/activity context	speed
universality	accuracy
easy to do	confusion matrix
cultural differences	precision, recall
physical/hardware constraints	axis of movement
relation to everyday actions	F-score
ergonomics	bias
subtlety	variance

This theme discusses some of the characteristics a prototyping tool could have to streamline different tasks for a designer. The previously discussed theme T8 of team communication and collaboration reflected upon some challenges the designer faces while developing prototypes. Moreover, while doing the design task, novices mentioned having trouble coming up with ideas for gestures or possessing limited bandwidth to reflect upon all the conflicting gestures which could lead to false positives. They also accepted explicitly that they didn't have the know-how to be able to prototype or implement the gestures. This calls for features in a tool which can provide examples of existing or commonly observed gestures to seed creativity and facilitate exploration. It could also provide flexibility for a designer with limited or no machine learning expertise to be able to effectively work with different sensor data or algorithms. The tool, in essence, could be an enabler that extends the capabilities of a designer.

3.4 Summary

Interaction designers play a key role in ideating, prototyping, and evaluating potential gestures for end-users of mobile and wearable devices. They are also the primary stakeholders of motion gesture prototyping tools. This Chapter introduced categories and themes in the motion gesture design process that emerged from the analysis of formative interviews with expert and novice designers. My formative work and literature review informed the creation of a design space of motion gesture prototyping tools, including a set of dimensions to position prior work and provide opportunities for future tool development. My findings led to an initial implementation of a designer-focused mobile gesture prototyping tool, allowing designers to create and test diverse gestures by demonstration in naturalistic settings. Future work includes additional features informed by my design space and extensive user testing.

CHAPTER 4

DESIGN SPACE OF MOTION GESTURE DESIGN TOOLS

Despite more than two decades of research, the use of motion gestures in commercial interactive systems beyond gaming consoles is still limited. I posit, based on the evolution and adoption of multi-touch surface gestures, that the lack of prototyping tools for interaction designers limits the design and evaluation of novel motion-based gestural interfaces. To address this situation, I synthesize prior research, review commercially available products, and draw insights from our own experience building recognition systems into a multi-dimensional design space of motion gesture design tools. I use a triadic framework consisting of gestures, designers, and tools to define and explain the relevance of each of these dimensions. I further illustrate the descriptive power of this space by situating prior work within it. Finally, I identify gaps and make recommendations for both industry professionals and academic researchers interested in bringing sensor-based interaction techniques, such as motion gestures, closer to end users.

4.1 Introduction

By a conservative estimate, the first demonstration of an in-air gestural interface was the "Put-That-There" system [13] in 1980, which showcased the integration of speech with pointing gestures. Today, almost four decades and hundreds of publications later, I have ubiquitous smartphones and wearable devices which are fully capable of supporting in-air gestural interfaces, but very few instances where this expressive input modality is utilized and promoted. Why do I still predominantly interact with our smart devices via multi-touch and speech input? I believe that there are several contributing factors.

First, although gestures are found to be "expressive, fast to access, and facilitated by ubiquitous inertial sensors" [107], they are not systematically explored and semantically categorized. While existing taxonomies focus solely on a few factors such as a specific device, a richer motion gesture vocabulary can only be developed by considering a wider variety of categorizations. For example, the intended purpose of a gesture within an interface is a valuable frame of reference when selecting suitable gestures from a library of gestures. An example of a missed opportunity is multi-device gestures.

The second factor that adversely affects the popularity of motion gestures is limited understanding of stakeholders interested in gestures. Most prior research concentrated efforts on providing guidance to developers while placing little emphasis on applying these findings for interaction designers. Even when researchers consider interaction designers, they are stereotyped as professionals lacking programming and pattern recognition expertise. While this is somewhat true, it does not seek to address or understand their processes, expectations, and challenges. In this chapter, I first provide a brief description of all the interested parties in gesture design. I then provide a richer description of considerations designer faces when designing gestures.

The final focus area that has resulted in the limited adoption of gestural interfaces is the lack of tools to design and prototype gestures. Back in 2000, Myers et al. correctly predicted that most of the interfaces will be off the desktop, hence tool designers should build support for new recognizer-based interaction techniques such as motion gestures [98]. Indeed a new wave of interfaces are being experienced on mobile and wearable devices, and within virtual reality environments, however, their development is still restricted to desktops that represent stationary and indoor settings. Since these devices are carried and used in varied contexts, a designer should be allowed to prototype and evaluate their designs in these naturalistic contexts. Hence, we need more mobile gesture prototyping tools. Furthermore, because there is a direct impact of the activity context on the sensor data (e.g., ContextType [44]), designers of gestural interfaces must be able to use such contextual information.

An exploration of the gesture interface design space would help to articulate a richer vocabulary for exploring gestures, a better understanding of designers needs, and to help shape the tools needed to design. There is a long history of design space articulations in the human—computer interaction (HCI) research. Back in 1990, Card et al. presented a design space of input devices [20]. Since then, HCI researchers have generated design spaces for many long-standing themes of research including input and interaction techniques [20, 57, 60, 122], design and prototyping tools [98], interactive applications and devices [31, 46, 102], user preferences [65, 78, 116, 133], and sometimes even whole generations of computing [2, 125]. These attempts to summarize an extensive survey of literature highlight the descriptive, generative, comparative, and prescriptive [9] purposes of a design space, thus laying the groundwork for future explorations. In this chapter, I make a modest attempt at covering the vast space of motion gesture research as it is applicable to design of new and better tools.

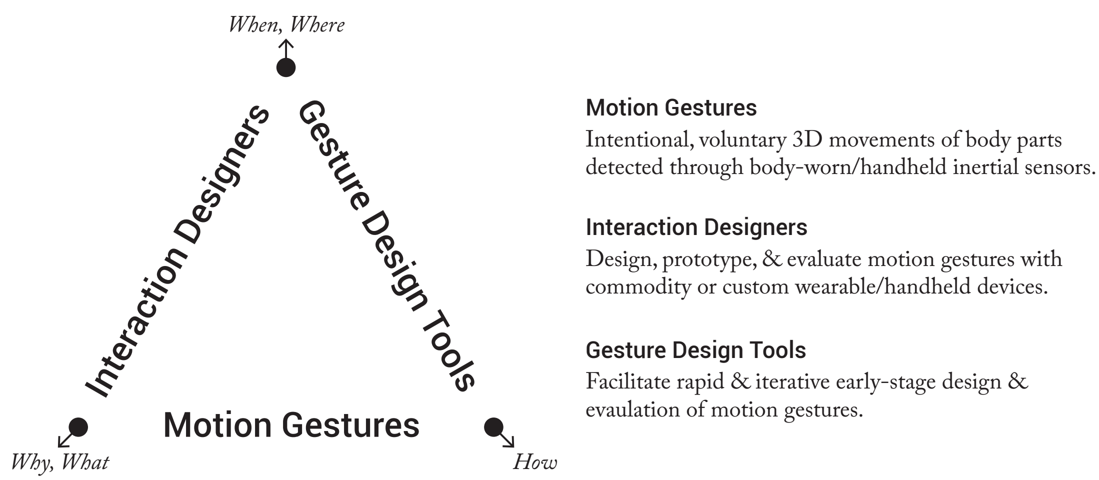


Figure 3: A triadic framework that visualizes the key entities in the gesture design process along its edges and opportunities that arise as a result of their interactions on the vertices. Each of the three entities are also defined in the right half.

I use a triadic framework (see Figure 3) comprised of gestures, designers, and tools as described above and in the next section to synthesize my personal experiences building gesture recognition systems, along with insights gathered from my survey of the literature and commercial products. Consequently, I have formulated a multi-dimensional design space to inspire further development of motion gesture design tools. In this chapter, I contribute a description of this design space. Additionally, I situate prior work within this design space to identify gaps and present opportunities for further development of such tools. Finally, I provide recommendations for both researchers and technology industry professionals who are interested in making motion gestures a viable and useful input technique for contemporary mobile and wearable devices.

4.2 Generating the Design Space

In the pursuit of a solution for any design problem, a *designer* iteratively works on an *artifact* through a *tool*. Likewise, motion gesture design can be modeled as interactions between at least three primary entities:

1. a gesture (artifact)
2. a designer, and
3. a tool.

Through this triadic framework (visualized in Figure 3), I will chart out a design space of motion gesture design and prototyping tools. In the process, I will describe different properties and categories of 3D motion *gestures*; challenges, processes, and considerations of *designers*; and the requirements and limitations of a prototyping *tool*. I begin by situating my review of existing literature and guidelines.

4.2.1 Recent Trends

Emergent trends in technology development and adoption influence the work of interaction designers and the design of their tools. First, there is a seismic shift towards mobility and always-available technology. As a consequence, new usage patterns have emerged. This is especially true for smartphones and wearable devices which are increasingly used on-the-go and accompany their user everywhere. Consequently, there is an increasing need for short-lived interactions (or *microinteractions* [7]) such as motion gestures. Moreover, increasingly powerful mobile platforms, backed by unlimited storage and processing capabilities through cloud services, can enable in situ and context-aware development.

Second, ownership of multiple devices with similar functionality and interoperability increases redundancy as well as the complexity of the interaction space. For example, a smartwatch can be used both as a companion to your smartphone and a standalone device.

Finally, most interactive technologies, including handheld, wearable (including virtual reality headsets), and ambient (e.g., Kinect sensor) devices, ship with embedded sensors which can be directly used or appropriated for interaction purposes. Hence, commercial as well as academic research efforts are dedicated to the expansion of novel input techniques to existing devices (e.g., smartphones, smartwatches) as well as prototyping of new interactive devices (e.g., a ring). Moreover, due to an availability of multiple input modalities (e.g., speech, touch, gestures) on any of the smart devices, designers can explore both multimodal and unimodal interactions.

4.2.2 Gesture-Based Application Design

In addition to current trends, I also anticipate that in the near future designers will be creating gesture-based mobile and wearable applications. Every human-computer interaction involves the performance of an action by a human followed by the perception of a change in an interface. For motion-based gestural interfaces, gestures are the actions and visual/auditory/haptic feedback in the interface is perceived. As such, there are three primary tasks for a designer of a gesture-operated application:

- **Designing the user interface**, which involves laying out content and interactive elements, typically on a visual canvas.
- **Prototyping motion gestures**, which comprises the identification and implementation of gestures that can be used as input to the user interface.
- **Mapping gestures to controls** of an interface.

I do not emphasize *the development of application logic* here because it is usually outside a designer's purview. Although there are equivalent tasks found in the extensive literature on graphical user interface design, the design of gesture-driven interfaces poses newer challenges. Norman and Nielsen support this assessment by highlighting that gestural interfaces do not comply with well-established interaction design principles and practices such as visibility, consistency, error recovery, etc. [103]. However, they only consider interfaces with a visual display and, moreover, do not touch upon challenges pertaining to prototyping of motion gestures and their mapping to functions on an interface. Note that gesture prototyping is the primary focus of this chapter, but for the sake of completeness, I will briefly touch upon challenges pertaining the other two tasks as well.

4.2.2.1 Challenges In Designing Interfaces

Supporting the design of user interfaces has been extensively researched, and is now mostly a commercial endeavor. There are advanced integrated development environments (IDE) for all major platforms (e.g., Android Studio for Android, XCode for iOS, etc.), which allow the development of user interfaces in tandem with programming logic. Using these programs, the UI is created through visual programming techniques such as drag-and-drop elements from a palette of components or directly manipulating underlying code. Because IDEs provide a high entry barrier for designers, they normally do early stage prototyping or wireframing outside IDEs in tools such as Sketch[1] which provide support ranging from static screens to interactive pages and in some cases allow testing on real devices. However, none of these advanced tools provide support for designing new input techniques, such as gestures, for triggering actions.

4.2.2.2 Challenges In Prototyping Motion Gestures

Even more so than 2D multitouch gesture interfaces, designing 3D in-air motion gesture interfaces requires a new paradigm. For instance, the social acceptability of performing gestures in public when the effects are not directly perceivable by bystanders is a concern [97]. In contrast, typing on the keyboard of a feature phone or tapping on a touchscreen even in social settings is acceptable. It is likely that an increase in the use of motion gestures will make them more acceptable, but which types of gestures could gain such initial acceptance is unclear. Another shift I will need is in thinking about new types of feedback and, perhaps, feedforward mechanisms [31]. As Bellotti et al. [11] speculate, I will have to “establish new non-GUI ways” to perform even the most basic operations such as cut, copy, paste or identify operations for which motion gestures are better suited. I am reminded of the following quote originally from Bill Buxton’s book titled Sketching User Experiences [19], and further adapted by Hinckley and Wigdor [60].

> “Everything, including touch, is best for something, and worst for something else.”

[1] `https://www.sketchapp.com/`

Another challenge is about the naturalness of gesture prototyping. Often tools developed for gesture recognition place artificial constraints on the process of training and testing gesture recognizers. Most commonly, recognizers rely on pre-segmented sensor streams produced through manual delimiters (e.g., using start and stop buttons) or artificially induced seemingly natural delimiters (e.g., starting and ending a gesture at a known rest state). Though these approaches have allowed researchers to build proof-of-concept systems which perform well in controlled settings, their ecological validity is questionable. For example, based on an observation of gestures performed during the speech, Wexelblat [132] raises concerns regarding "issues of coarticulation" or overlapping gestures.

4.2.2.3 Challenges in Mapping Gestures to Controls

User interface designers have at least four common input modalities to choose from: physical buttons, touchscreen, speech, and gestures. Since each of them presents different implicit constraints, careful consideration must be placed on selecting the right input technique for the job. Moreover, given an input technique, mapping of a user's action to an interface command can be challenging, especially for a new modality such as motion gestures. One may ask whether to use semantic gestures or random motions. Furthermore, should it change based on the context of use? Moreover, gesture recognition may impose constraints such as the use of gestures as discrete input, which makes them cumbersome to use for continuous controls like knobs and sliders.

4.3 Gestures: Considerations & Categorizations

What gestures does a designer want to be able to design? What types of gestures should a system detect? What are the attributes of a gesture? What is unique about motion gestures?

4.3.1 Definition & Sensing

I define motion gestures as *intentional, voluntary 3D movements of body parts detected through body-worn inertial sensors.* For the most part, I consider symbolic gestures which are "standardized gestures, complete within themselves, without speech" [132]. I interpret the property of being *body-worn* in a loose sense where the sensors could be directly strapped to, clipped on, or held by a body part or carried in clothing or accessories such as a purse. However, I exclude interactions that are performed on a static surface (e.g., a table) adjacent to a sensing device as exemplified in BeyondTouch [138]. But I do include interactions on the device such as taps on the back of a phone [139] and on the skin around, for example, a smartwatch as in TapSkin [137].

I rely on inertial sensing because of its practicality, affordability, and availability. Inertial sensors (e.g., accelerometers, gyroscopes) are ubiquitous in handheld and wearable devices today. Their lightweight and small form factor allow them to be strapped to a body part to provide direct sensing of its motion. In the research literature, they have been utilized for detecting taps on and around a device [138], coarse motions such as writing in the air [4], or motion states [80] such as walking, running, or driving to infer interaction needs. In the commercial domain, they have been embedded in activity recognition and tracking devices (e.g., Fitbit), gaming controllers (e.g., Nintendo Wii), and other commodity devices. Lately, I have begun to see inertial sensing leveraged for motion gesture input with a growing set of gestures, available on Android smartwatches.

Next, I suggest various dimensions along which gestures could differ, except for the sensors used.

4.3.2 Expected, Sensed, & Desired

Benford et al. [12] propose a 3-dimensional framework to guide the design of movement-based interactions such as motion gestures. They invite interaction designers to think about *expected*, *sensed*, and/or *desired* movements.

Expected movements include natural movements that a human body can perform comfortably, and repeatably with all the variations in speed, degrees of freedom, range, and accuracy.

Sensed movements refer to motions that a given sensor can detect reliably. Depending upon the accuracy and resolution of a sensor, this might exclude some of the expected movements while including hypothetical movements that a human cannot perform.

Desired movements emphasize the requirements of an application. In the best case scenario, this includes only possible motions (expected and sensed). Sometimes, it may not. An illustrative example from the article is a players desire to physically fly to control their avatar in a 3D game.

4.3.3 What Body Part(s)?

Based on my definition of motion gestures, the first differentiating factor is: the body part used to perform a gesture. Theoretically, I can use any body part as long as its motion is repeatable and interesting, however, practically I see the arm, hand, and head motions to be the most common. Degrees of freedom, the range of motion, accuracy, and stability of motion vary with the selected body part [12]. Handhelds such as smartphones and tablets enable coarse arm movements like writing in the air [4] and also relatively smaller tapping gestures on the device, as noted above. Smartwatches are particularly good for the arm, hand, and even finger motions [21, 130] while also enabling novel on and around watch tapping interactions [137, 140]. Moreover, head mounted devices such as Google Glass can be used for detecting head motions (e.g., nods, side bob) in humans and dogs [128]. Finally, I have also seen a few examples of foot gestures with custom sensors placed directly on foot [29] or indirectly through the phone mounted inside a pocket [121]. An opportunity to extend the current vocabulary of gestures is to consider gestures involving multiple body parts as enabled by multiple devices people already carry.

4.3.4 Temporal & Spatial Considerations

Gestures can vary along temporal and spatial dimensions. Spatially, gestures can vary from coarse (e.g., waving arms in the air) to fine-grained motions (e.g., raising an eyebrow). Temporal characteristics include the speed and duration of a gesture. Speed and length of a gesture naturally vary from one sample to another due to fatigue, physical variations such as arm lengths, as well as distractions. While these slight variations are typically useful for building a better representation, in contrast, bigger, intentional variations are better treated as a different gesture altogether much like differences in the swipe and rapid flick motions on a touchscreen. Related to speed and duration is the notion of repetitive gestures in which a small gesture is repeated multiple times consecutively, often to increase the likelihood of detection and decrease errors.

4.3.5 Coverbal or Autonomous

Researchers have proposed several classifications of gestures based on the purpose of a gesture (see Wexelblat's review of prevailing taxonomies [132]). For example, gestures can be categorized into semiotic (that communicate information), ergotic (that manipulate physical objects), or epistemic (that explore the environment for information) [60]. However, Bolt's "Put-That-There" [13] system brought forward a distinction that gestural interaction designers and researchers should consider; whether a gesture is autonomous, meaning standalone without speech, or coverbal, that is, co-occurring with speech.

4.3.6 Unimodal vs. Multimodal

Motion gestures can be used in a unimodal or in a multimodal fashion. Besides motion gestures, other common gestural interfaces can be characterized as touch- [133], pen- [56], tangible- [119], or camera-based (or in-air) [22, 23, 124]. Combinations of these modalities with motion gestures (multimodal) have also been explored (e.g., pen with motion [55], touch with motion [59]). In the multimodal case, motion gestures can be used before, in-between, or after the use of other interaction technique(s) [25]. Motion gestures can also be used in conjunction with speech and non-speech acoustic input (e.g., Whoosh [112]), as well as physical button input.

4.3.7 Discrete vs. Continuous

Gestures can also be characterized as *discrete/isolated* or *continuous* [16, 52, 132]. A discrete gesture has a clear beginning and an end in time, whereas a continuous gesture has no clear delimiters. For example, waving your hand is a discrete gesture which can be clearly isolated in time. However, playing a musical instrument may involve a series of movements with no clear start or end.

4.3.8 Atomic vs. Compound

A gesture can also be defined as *atomic* or *compound*. Atomic gestures are those that cannot be further decomposed, and which can be combined to generate other gestures. A compound gesture consists of a linear combination of atomic gestures. For example, from the previous example, waving your hand is a compound gesture that consists of two atomic gestures (i.e., wave right and wave left) in rapid succession.

4.3.9 Purpose: Response, Command, or Activation

Professional interaction designers can consider the following categorization of gestures while designing an interface to a device and applications on it. Furthermore, this categorization can inform the design of novel interaction patterns that can enhance the end-user experience with gestural interfaces. The gesture types presented below vary by the purpose of use, the frequency of use, expressivity, and expected error rates.

Response gestures are performed in response to a system-initiated notification. After the notification arrives, the system can switch to gesture listening mode for a predefined duration. Upon timeout, the notification is still accessible but the user has to use non-gestural means to retrieve it. After selecting a notification, an application might take over user's input in which case the system is waiting for command gestures.

Command gestures are used for operations within an application, including special scenarios when an application is running in the background with a limited palette of controls exposed. Command gestures need to match the needs of an application designer. Unlike multitouch gestures, a small set of motion gestures will not typically suffice for the varying needs of each application. However, if interfaces are custom designed for motion gestures, I can expect standardization of interaction patterns for gestural interfaces, thus motivating a small set of common command gestures that work across applications.

Activation gestures are globally available gestures for either user initiation and termination of gesture listening mode. Command and response gestures can be safely performed after activation. Activation gestures may also be used for triggering the user's most frequent shortcuts, such as opening your favorite application. Note that by necessity activation gestures demand low false positive and high true positive rates. As a result, they are typically a small set of gestures designed specifically to minimize accidental triggers often at the cost of ease of learning. Ruiz and Li's DoubleFlip [115] is one such gesture.

Table 4: Characterizing the different target stakeholders in the gesture design process: end-user, designer, developer, and researcher. (NR = Not required)

Stakeholder	Experience			Requirements From Tool
	Coding	*Machine Learning*	*Motion Gesture*	
End-user	NR	NR	NR	Use gestures
Designer	None-Low	None-Low	None-High	Prototype gestures
Developer	High	Low-High	None-High	Develop recognizers
Researcher	Intermediate	Low-High	Low-High	Prototyping and user study support

4.4 Gesture Designers: Process & Challenges

Who is interested in designing motion gestures? What process does a gesture designer follow? What are the challenges a gesture designer might face?

4.4.1 Types of Stakeholders

There are four potential users in a gesture design process: (a) Interaction designers; (b) Developers; (c) Interaction researchers; and (d) End users. Though all of these are designers at some level, I identify them on the basis of their goals and varying levels of expertise in coding, machine learning, and motion gestures (see Table 4).

4.4.1.1 Interaction Designers

Interaction designers are the primary stakeholders for a prototyping tool. Their objective is to identify an appropriate set of gestures informed by the considerations discussed earlier, rapidly prototype, and evaluate them with potential users. Hence, a tool could support these primary tasks, allow programming by demonstration, and handle the complexity of building a recognizer.

A further distinction could be made between designers on the basis of their technical capacity. I consider designers residing in big technology corporations with a good understanding of computing technology and a fair knowledge of development as *high technical capacity designers*. On the other hand, designers at design firms or small design teams within corporations wanting to explore a few simple gestures for their one-off projects, are categorized as *low technical capacity.*

4.4.1.2 Developers

For *developers*, the goal is to develop a robust gesture recognizer. A tool could allow the developer to experiment with different classifiers, features, and combinations of sensor channels. Furthermore, she may export data for further analysis on desktop-based tools (e.g., MATLAB). MobileWeka addresses some of these developer needs on the mobile phone [88], but does not support live sensor data collection and recognition.

4.4.1.3 Researchers

Interaction *researchers* like me can be considered a hybrid between a designer and a developer. They invent novel interaction techniques such as motion gestures, demonstrate their usefulness through sample applications, and evaluate them in user studies. They create proof-of-concept implementations and, in the process, end up creating makeshift tools to support their rapid, iterative process. For exploring motion gestures, they might need to perform data acquisition, visualization, and classification, as well as user evaluation. Because the engineering effort required to build a custom tool is not always justified, they desire robust yet flexible tools. However, the status quo is typically for each interaction research team to use a custom suite of tools. Hence, there is a need for shared tools and understanding to allow newcomers to jump right into this exciting area of research.

4.4.1.4 End Users

Lastly, I define *end-users* as people who will ultimately use the gestures. For end-users, the objective is to map input actions to useful applications, similar to IFTTT [2]. End-users can leverage motion gestures for fast access to common shortcuts on their phone or connected devices.

4.4.2 Designer Workflow

Interaction designers typically follow a four-step user-centered design process: gather requirements, design solutions, prototype, and evaluate (explained in detail by Rogers et al. [113]). For the case of motion gesture design, I anticipate a similar process starting with an

[2] `https://ifttt.com`

understanding of the situations and tasks for which gestures are desired, and ending with an evaluation, in situ or otherwise, of gestures prototyped with a tool. However, in the motion gesture design tool literature, I notice a three-step process consisting of design, test, and analyze phases [6, 52], inspired by Klemmer et al.'s [74] work. Often an additional step is added which addresses either mapping of gestures to application functions [8], as well as exporting gesture recognizers [52] or programming code [69] that developers can leverage to build gesture-based applications.

4.4.3 Authoring Approach

"Historically, there are two distinctive approaches for empowering developers to create complex interaction behaviors: programming by declaration and demonstration" [94]. *Programming by declaration* uses a high-level specification language to describe interaction behaviors. On the other hand, *programming by demonstration (PBD)* focuses on providing examples of the target interaction behaviors and is commonly used in interactive machine learning. The authoring of gestures can be carried out within the context of an application or *in situ*, as well as *out of context* in an artificial setting. The advantage of authoring in situ is that it enables the designer to uncover nuances and considerations of using particular gestures.

4.4.4 Evaluation Context

There are two ways a gesture recognizer can be evaluated and experienced after it has been implemented. A *standalone* evaluation means that the recognizer is evaluated within the prototyping tool itself and gestures are manually segmented. Secondly, the gesture recognizer can be *linked to particular applications* and evaluated in context, meaning running in the background and automatically segmenting and consuming gestures. Typically, a standalone evaluation is used for evaluating a variety of gestures without a specific application in mind. The second approach of evaluating in-context allows the designer to evaluate the gesture recognizer performance, as well as the effectiveness and appropriateness of the gesture during application use. Additionally, there are two ways a gesture recognizer could be linked to applications. A weakly linked scenario suggests exporting the recognizer code as a standalone application or as a built-in device system feature and using it to trigger events in applications. A strongly linked scenario requires integrating the recognizer directly into the source code of a new application and considering gestures to be used during application and interface coding.

4.4.5 Development & End-Use Environment

Abowd recently highlighted that the "programming environments of the personal computing generation consist of interactive development environments that exactly match the characteristics of the end user experience" [1]. In contrast, the user experience for mobile and ubiquitous computing "is the 3-dimensional physical world. However, the dominant development environment remains the 2-dimensional graphical user interface." For instance, a mobile authoring tool would be most appropriate for building interactions in mobile and wearable settings, where the end-user experience happens.

4.4.6 Data Collection

There are multiple ways of collecting sensor data, either *laboratory-based* or *in-the-wild*. A laboratory study, which may be useful early on in the design phase, is a controlled environment where the designer is able to experiment by constraining factors that would otherwise be difficult to control with in-the-wild or longitudinal data collections (e.g., external noise). On the other hand, more diverse and realistic use cases and observations may also be of interest at a later phase of design. CrowdLearner proposes two strategies for collecting training data in-the-wild: *participatory* and *opportunistic* sampling. Participatory data collection allows the tool to "learn when I ask the user to do something" wherever they are. Opportunistic data refers to the ability of the designer and tool to "learn when something interesting happens" [3].

4.5 Gesture Prototyping Tools

In what ways can a tool support the design and development of motion gesture interfaces? What are the features and components of such a tool?

4.5.1 Literature Review of Tools

Motion gestures are non-trivial to author because they require a designer to comprehend in what ways the human body can move, how does a sensor work, what gestures would be appropriate, memorable, easy to learn and use. The notion of reality-based interaction captures these considerations into a framework consisting of naïve physics, body awareness, social awareness, and environmental awareness [66].

Several researchers have tried to lower the barrier for developers who are not machine learning experts. Rubine's *GRANDMA* [114] is probably the first toolkit that allows for development of gesture-based applications. It lets a developer specify unistroke gestures, map them to interface elements, and configure the effect of gestures on them. Gillian and Paradiso's *GRT* [43] and Lyons et al.'s *GART* [95] provide real-time machine learning and gesture recognition support for gesture based applications through standard implementations for several algorithms. They are both modular. GRT is also available as an addon, among other such addons, for *openFrameworks*[3], "an open source C++ toolkit for creative coding." Westeyn et al.'s GT^2k [131], a precursor to GART, extends support for Hidden Markov Model (HMM) [110] implementations available in a speech recognition toolkit, called *HTK*[4], to gesture recognition. Fiebrink's *Wekinator* [40] is a tool with a graphical interface, which uses the Weka library [50], built for musicians and creative coders. It enables on-the-fly machine learning using various input sources, such as gestures and joysticks, to generate output fed to an audio synthesis instrument.

In contrast to toolkits that support multiple algorithms, several researchers have developed easy to implement, and computationally cheap gesture recognizers by either using rule-based or template-matching (e.g., Dynamic Time Warping, Nearest-neighbor) approaches for identifying simple features that are fast to calculate and have highest discriminating power. Wobbrock et al.'s *$1 recognizer* [134] is such a recognition algorithm for unistroke gestures that relies on trigonometric and geometric calculations. Li's *Protractor* [85] outperforms $1 recognizer in speed and savings in memory while providing rotation invariance.

[3] `http://openframeworks.cc/`
[4] `http://htk.eng.cam.ac.uk/`

Kratz and Rohs' *$3 Recognizer* [76] and *Protractor3D* [77] respectively extend the previous two methods pertaining to touchscreen gestures to the world of 3D acceleration-based gestures.

A different approach to building motion gesture recognizers has been identified by Amini and Li's *CrowdLearner* [3]. CrowdLearner proposes a novel framework for developers which leverages crowdsourcing (i.e., Amazon Mechanical Turk) to rapidly generate mobile recognizers for specific tasks and gestures, minimizing the overhead of recruiting participants and performing data collection in person.

In contrast to the work mentioned above, the tools reviewed below provide support for a variety of sensor-based interaction techniques for designers, who may not even have programming skills. Klemmer et al.'s *SUEDE* [74] is prototyping tool for wizard-of-oz speech-based interfaces that provides a simple design-test-analyze process. Long et al.'s *GDT* [90] and *Quill* [89] provide support for designing pen-based gesture recognizers and improving their designs through feedback and unsolicited advice respectively. Kin et al.'s *Proton++* [71] and *Proton* [72] support authoring of multi-touch gestures declaratively through tablatures, a novel graphical notation denoting the sequence of touch events over time, and regular expressions. In contrast, Lu and Li's *Gesture Coder* [93] allows authoring of multi-touch gestures for tablets using programming by demonstration. In a related system called *Gesture Studio* [94], they combine both demonstrative and declarative approaches within a video-editing metaphor and even allow exporting of source code that a developer can use.

While all of the other tools that I discuss in this chapter attempt recognition of a gesture only after it is completed, Zamborlin et al.'s *GIDE* [136] provides support for continuous real-time tracking of motions and synchronous feedback from motion recognizers for a variety of sensors.

Several tools emphasize visual authoring of gestures or classifiers. For instance, Fails and Olsen's *Crayon* [38] developed an interactive machine learning approach which allowed a user to iteratively train an image-based classifier by manually *painting* labels over portions of images. Baytaş et al.'s *Hotspotizer* [8] leverages declarative marking of discretized space elements, called *hotspots*, around a user's body for defining a gesture's trajectory to be detected by the *Kinect* sensor, a depth sensing camera that tracks skeletal motion. It also allows mapping of gestures to application functions through a graphical interface. Kim et al.'s *EventHurdle* [69] is an authoring tool for gestural input from a camera, hand-held sensors, and touch surfaces. The tool allows visually defining and modifying gestures through an interactive workspace and graphical markup language and generates programming code that represents the created gestures. In contrast to Hotspotizer and EventHurdle, *Exemplar* [52] and *MAGIC* [6] show visualizations of raw sensor data, which might not be easiest to understand by designers.

Hartmann et al.'s *Exemplar* [52] introduced the design-test-analyze workflow for authoring sensor-based interactions. Through a direct manipulation interface, it enabled novices to rapidly explore gestural interaction with a variety of sensors. Ashbrook and Starner's *MAGIC* [6], a desktop tool for the creation, and evaluation of motion-based gestures with a wrist-mounted 3-axis accelerometer, built on Exemplar in two ways. First, it exposed the machine learning (ML) measures of inter- and intra-class comparison scores and confusion matrices in a graphical representation. Second, it introduced a comparison of intended gestures against a repository of everyday hand motions in the analysis stage.

Now I turn to specific instances of mobile tools for designers. Kim et al.'s *M.Gesture* [68] is a mobile tool for visually authoring acceleration-based gestures with multiple devices. It uses a combined demonstrative and declarative approach in which acceleration data is visualized using a mass-spring metaphor and then lines intersecting this trajectory (similar to EventHurdle [69]) are placed to define a gesture. Kim et al.'s *CompositeGesture* [70] is similar to M.Gesture except for the absence of mass-spring metaphor and inclusion of support for mapping gestures to application functions. Finally, *Mogeste* [107], a tool I built, enables interaction designers to rapidly prototype and evaluate motion gestures *in situ* with accelerometers and gyroscopes.

4.5.2 Platform Choice

Choosing a target platform for the motion gesture prototyping tool is a key decision to be made based on the designer's goals, target devices, and envisioned application scenarios.

Most rapid prototyping tools in the literature are desktop-based tools [6, 34, 52, 74]. A *desktop* (or laptop) development environment provides a large screen, establishes multiple wired and wireless connections, and more importantly, leverages mature development and visualization tools. However, the form factor of the development equipment and operation in its proximity prevents designing in a wider variety of naturalistic and *ad hoc* settings, particularly for mobile and wearable sensor-based interactions [1].

Mobile platforms afford portability, boast higher penetration and daily usage, and provide simple direct manipulation interfaces [64]. Moreover, experience with a myriad of productivity applications on mobile devices today shows that small screens can be used effectively for rapid design and exploration. In fact, both M.Gesture [68] and Mogeste [107] are early examples of mobile motion gesture design tools that motivate future development in this area.

As wrist-worn and head-mounted *wearable* platforms become more powerful and accessible, I expect a new category of tools to arise that will leverage their always-on, always-available nature as well as embedded sensors. But due to their small screen size, short battery life, and limited computation power compared to other platforms, I hope these tools will be extremely simple and provide support for a few critical features. Two recent examples of recognition systems running on commodity smartwatches for non-voice acoustic interaction [112] and finger motion gestures [130] are very encouraging.

Finally, *hybrid* solutions describe functionality distributed across wearable, mobile, desktop, and even cloud platforms (e.g., CrowdLearner [3]). The advantage of hybrid solutions is that processing may be offloaded to a more computationally capable cloud server or remote device. However, in many cases, using a hybrid solution assumes continuous connectivity to the remote device. As seen in recent examples [93, 94], these hybrid solutions mainly focus on using the mobile device as a sensor platform alone.

Figure 4: Typical machine learning pipeline.

4.5.3 Machine Learning Back-End

There are two approaches to building a machine learning back-end or pipeline (like Figure 4) in a motion gesture prototyping tool: a white-box or a black-box.

A *white-box* exposes all functionality of the pipeline, allowing the user (designer in this context) to tune parameters, select features and sensors, and experiment through trial and error. Although the white box approach adds flexibility to the system for user-configurations, it will only be useful for designers or creative explorers/coders either with basic knowledge of machine learning to experiment with the models based on their particular application or with a willingness to explore the tool and attempt at trying different models.

Conversely, a *black-box* approach hides all details about how raw sensor data is pre-processed, models are trained, and algorithms are parameterized. A black-box tool could automatically run experiments in the background to find the best set of features and algorithm to use given training data. It would be appropriate for a designer who wants to take advantage of machine learning but is not interested in understanding the details. Although this paradigm would work for pre-built models and features, it might not be flexible enough to work for all use cases.

In either approach, it would be helpful to communicate different options and tool features in a designer-friendly language to optimize usability.

4.5.4 Recognizer Feedback

The design process involves a feedback loop between the tool and the designer. The tool provides feedback to the designer about its state, overall recognizer accuracy, and recognized testing gestures. The designer can respond to the feedback by collecting additional data, removing gestures affecting performance, or additional steps through trial and error. The measures of *inter-class* and *intra-class* comparison scores can be used to guide the designer on the uniqueness and reliability of a gesture. *Confusion matrices*, if communicated effectively, can be useful for designers during evaluation, once an entire training set has been collected.

4.5.5 Gesture Representation

Visual representations of sensor data can reinforce not only memorability and learnability of gestures [67] but also cognizance of what gesture has been performed. There can be two ways to represent gestures:

- *Static Representation*: a textual description of the gesture, a photographic or symbolic image of the gesture, or a static visualization of the raw sensor data.
- *Dynamic Representation*: 2D/3D trajectories of gestures in free space calculated from raw sensor data, gesture demonstrations captured with videos, or animations of a human figure using an inverse kinematic model.

4.5.6 Interaction Modalities

Depending on the designer's context and application needs, she may want to interact with the prototyping tool and receive feedback using different modalities. *Input modalities* to interact with the tool include the physical/virtual keyboard, touch, gestural, and speech input. Touch is suitable for situations where the user engages directly with the visual screen. Gestural and speech modalities allow the designer to interact with the tool in an eyes-free manner, facilitating situations where the user is mobile or distracted. Output feedback provides the designer with state information of the tool, as well as the performance of the recognizer. *Output modalities* include auditory, haptic, and visual feedback. These modalities can be useful as feedback during in-context usage and about connected devices.

4.6 Types of Studies & Evaluation

So far our discussion has centered around the development of tools to support designers in their discovery, creation, and evaluation of motion gesture designs. It is also important to understand various methods researchers employ for both formative data gathering and summative evaluations. Workshops and elicitation studies are good examples of the former, whereas first-use and long-term-use studies along with workshops are useful for evaluating a tool. In contrast to the user-centric approaches mentioned before, the evaluation of gesture recognizers' performance is of value for more technical contributions.

4.6.1 Elicitation Study: A Method for Brainstorming Gestures

When exploring a new interaction technique, or use of an existing technique in a new scenario, researchers rely on a crowdsourced brainstorming approach in which target users are asked to envision gestures they would use for a reference task. Wobbrock et al. [133] first used this concept for eliciting gestures for interactive tabletops. Since then researchers have applied the method of elicitation studies in the formative phases for finding emergent, user-defined gestures (for example, smartphone [116], smartwatch [5], multi-device scenarios [78, 62], single-hand microgestures [21]) which typically result in taxonomies.

It is also possible to do the opposite, where a reference set of gestures is provided with detailed description and users are asked: "what would you use this gesture for?" This strategy to elicit natural associations of gestures to a task can be useful in defining standards and conventions.

4.6.2 Evaluating a Recognizer's Performance

In order to support their claims of contributing a viable gesture recognition system, researchers often provide results of its evaluation on established metrics such as precision, recall, accuracy, etc. I refer readers to Bulling et al.'s [16] tutorial on activity recognition systems, Ward et al.'s [129] discussion on performance metrics, and Domingos' [36] collection of good advice from machine learning experts for a comprehensive treatment of the topic.

In accordance with the mentioned articles, I note that lab-based, offline evaluation of recognition systems, though a good starting point, doesn't say much about its performance in user-driven and out-of-lab testing. Moreover, sometimes the flexibility of a classification algorithm in recognizing different types of gestures outweighs with a reasonable accuracy is more valuable than high accuracy on a small set of gesture, especially when building a general purpose tool. Therefore, researchers mostly fall back on simple known techniques such as Dynamic Time Warping (e.g., Exemplar [52], MAGIC [6], M.Gesture [68]), or more advanced techniques with proven history of good performance and wider applicability such as Hidden Markov Models (e.g., GART [95], GIDE [136]), and Support Vector Machines (e.g., Wu et al. [135], Mogeste [107]).

4.6.3 Workshop

Once a tool has gone through several iterations, workshops are a great way to get a large number of users to try out your tool in an uncontrolled setting. Workshops can be conducted within formal settings such as conferences, classrooms often with homogeneous populations, or in informal settings such as community centers with heterogeneous populations. The main goals of a workshop are to test the ease of learning, use with minimum instructions, and to assess the opportunities a tool opens by lowering barriers to entry and to advance the ceiling. Workshops are also a great forum for encouraging collaborations, collectively reflecting on the role of a tool and brainstorming new avenues of research.

4.6.4 First-Use Study

A first-use study allows a researcher to test the usability of their tool by recruiting first-time users and letting them perform tasks with it. Brush [79] refers to this type of user testing as proof of concept field study. For example, Hartmann et al. [52] recruited 12 HCI students for completing prototyping tasks in a stipulated time. In the case of Exemplar [52], first-use studies are typically conducted in controlled environments to allow recording of participant's actions and reactions. A potential drawback of this method is the lack of ecological validity. Moreover, the presence of the researcher who is also a developer of the tool can lead to over-reliance on his/her knowledge.

4.6.5 Long-Term-Use Study

An alternative and subsequent method is an evaluation through long-term use of a tool in naturalistic settings, for open-ended tasks. This method aligns with Brush's [79] third type of field study: Experience using a prototype. The primary goal of such extensive testing is to provide real evidence for the utility of the tool and also to learn about emergent behaviors. An obvious challenge in this approach is a lack of control, hence finding ground truth is not always possible. However, application logs and regular experience sampling can lead to the collection of rich qualitative and quantitative data.

4.7 Considerations For Industry & Researchers

4.7.1 Enabling Development & Use of Gestural Interaction

4.7.1.1 SDK

A software development kit (SDK) is a set of development tools provided by the developer of a target hardware/software platform to allow others to create applications for his/her platform. For example, the Android SDK allows developers to create applications for devices running the Android OS. To enable development of gestural interfaces, I propose providing support within an SDK for implementation, simulation, and debugging.

4.7.1.2 API or Library

An application programming interface[5] (API) "is a set of subroutine definitions, [communication] protocols, and tools for building application software." A library is just one instance of the implementation of this interface which is tied to a programming language, or a platform. With standard libraries for supporting common tasks involved in the implementation of a gestural interface on a specific platform or with platform agnostic web-based APIs, developers can easily implement gestural interfaces. However, designers will need more visual means of using these APIs, which is where an IDE comes in.

[5] `https://en.wikipedia.org/wiki/Application_programming_interface`

4.7.1.3 IDE or Plugin

Every major mobile OS comes with its own integrated development environment (IDE) to support the development of all aspects of application software including user interfaces and back-end logic. Though support for UI design through drag-and-drop like visual programming metaphors has become better over the years, no support exists for designing gestural interfaces. Until major platform developers release such support, I foresee the development of plugins that will allow a gestural interface development team to conceptualize and implement for this emergent interface paradigm.

4.7.1.4 Suite of Tools

Although an IDE is ideal for development, often a disintegrated approach of providing a suite of tools instead can help with separation of concerns. As opposed to an IDE, which can get unmanageable and provide a high entry barrier, individual tools for targeting specific aspects of an interface such as designing new gestures can provide low threshold yet high ceiling. However, for such an approach to work, importing to and exporting from each tool within a suite should be properly managed to allow seamless integration of various pieces into a single whole.

4.7.1.5 End-User Customization

While consumers are well accustomed with using only input methods and interaction techniques provided by the developers of a platform, motion gestures are a special case which may require allowing calibration and customization by end-users. After all, they are supposed to use it everywhere, and a poorly designed gesture or a well-intentioned but culturally inappropriate gesture might cause frustration. Moreover, as I have seen over and over again with problems discovered by use of touchscreen gestures, a designer cannot foresee all usage patterns. Therefore, along with a few subtle, default set of gestures that work magically, a platform should provide options for users to choose from a repository of gestures, which they can download, try out, calibrate, and link with their favorite operations. Bragdon et al.'s GestureBar [14] as a gesture learning and practice tool for gestural interfaces and Google

Gesture Search application for the Android platform as a custom gesture-based shortcut creation and practice tool are good examples of this paradigm, but both target only 2D gestures.

4.7.2 Application Areas

So far I have only discussed motion gestures as a novel input method to computing devices. But now I shift my attention to listing potential target domains where motion gesture recognition and analysis can be advantageous.

4.7.2.1 Everyday Interactions

As discussed so far, motion gestures when supported on commodity computing devices can be a useful alternative to the dominant input methods of touchscreen and speech. There are already a few arm/wrist gestures that work out of the box in Android Wear, as well as Apple smartwatches. Google Glass also supported a few experimental head gestures.

4.7.2.2 Gaming

The Gaming industry has always been a harbinger of new interactive technologies. For instance, in 2006 the Nintendo Wii was released with a gaming controller that could sense its in-air movements using accelerometers. It not only entertained users of all age groups, it enabled researchers (e.g., Schlömer et al. [119]) to explore motion gestures as a viable input modality. Many years later, with increasing popularity of virtual reality, we are seeing a resurgence of interest in in-air motion gestures not just for gaming, but even as the dominant input modality. I see an interesting opportunity in using existing commodity wearable and handheld devices as input controllers within VR environments.

4.7.2.3 Assistive Technology

Dr. Stephen Hawking's personal assistive technology developed by Intel allows him to interact with a computer via cheek movements. This technology is a prime example of how gestural interaction can be utilized to enable any functional body part as a means to interact with computing technology. Increasingly smaller form factors of inertial sensors also allow development of custom devices for tapping into subtle, controllable motions such as raising an eyebrow. In another example, researchers have found inertial sensors placed in an around-ear headset to be useful for detecting eating by analyzing jaw motions [10].

4.7.2.4 When Touching is Risky or does not Work

The ubiquity of smartphones and wearable devices encourage their use even in situations where touching them is either dangerous or doesn't work. For instance, if a physician during surgery wishes to refer to information on a head-mounted device or a desktop computer, she should refrain from touching these devices due to a risk of infection. Similarly, a worker in a food processing plant shouldn't touch his watch or phone. Besides the risk of contamination, there are also concerns about a touchscreen being inoperable if your hands are wet or covered in dirt, a likely scenario for a farmer trying to browse information about his crop. Lastly, sometimes you just need to perform a quick action, for example, if a worker on assembly line needs to check a manual just-in-time for the next step. Motion gestures can be quite helpful in these scenarios.

4.7.2.5 Motion Tracking & Analysis Scenarios

So far I have looked into scenarios for which purposeful gestural interactions are designed to operate a computing device. Let's now briefly look into following scenario [6] where tracking and analysis of motions, intentional or unintentional, is important, not as an input to a computing device, but instead for understanding and correcting motions.

- **Workforce:** An employer might want to continuously monitor the motions of their workers to help them improve technique, avoid injuries, and be more productive.
- **Therapy:** A chiropractor might want their patients to exercise regularly and correctly even while working out at their home.
- **Physical Training:** A gym instructor or a personal trainer might want their client to easily record their reps while also receiving detailed feedback on their form.
- **Sport:** Similar to the above scenarios, a sports trainer can record, track, and analyze motions of several players at once to draw comparisons, produce insights, and prevent

[6] List based on `http://focusmotion.io`

injuries from over training.

4.7.2.6 *Retail Opportunities Driven by IoT*

The Internet of Things (IoT) is a recent trend which postulates a future in which every object is digitally connected. It naturally leads to ideas of smart objects followed by smart homes, and ultimately smart cities. In this paradigm, marked by ubiquitous connectivity and abundance of sensing technologies, manufacturers of consumer products are open to trying unconventional interaction techniques. As a suggestive list, toy, furniture, and home appliance manufacturers are well positioned to make use of motion gestures as a novel, playful input modality.

4.7.3 Releasing Data Sets & Custom-Built Tools

My review of tools cites more than 20 pieces of exemplary work, but, ironically, I don't have access to any of those tools. It is unfortunate that in user interface research, the time and engineering effort it takes to build, test, document, and release a robust tool is overwhelming, and not typically considered a research contribution [63]. Moreover, interaction researchers do not release any datasets they might have collected during a technical evaluation or a user study, leading to wasted efforts of collecting similar kinds of data. In the light of this observation, I encourage researchers to release their custom pipelines, tools, and data sets even if they are not fully documented.

4.8 Summary

Interaction designers play a key role in ideating, prototyping, and evaluating potential gestures for end-users of mobile and wearable devices. They are also the primary stakeholders of motion gesture prototyping tools. This chapter introduced categories and themes in the motion gesture design process that emerged from the analysis of formative interviews with expert and novice designers. My formative work and literature review informed the creation of a design space of motion gesture prototyping tools, including a set of dimensions to position prior work and provide opportunities for future tool development. My findings led to an initial implementation of a designer-focused mobile gesture prototyping tool, allowing designers to create and test diverse gestures by demonstration in naturalistic settings. Future work includes additional features informed by our design space and extensive user testing.

CHAPTER 5

MOGESTE: INVESTIGATING IN SITU MOTION GESTURE DESIGN

Based on the formative investigation, I realized that motion gestures are very different than the contemporary input modality of multi-touch because unlike screen-based multi-touch interactions, 3D motion gestures can be seen as a public performance. Therefore, getting out of the lab and designing in naturalistic contexts is critical to their success. Moreover, because I settled on body-worn inertial sensors for gesture detection, any motion of the body part we are designing for will inadvertently affect the sensor readings. Due to the very nature of wearable devices which we carry everywhere, unintentional motion is unavoidable. Hence, capturing representative data in a variety of settings is essential for building robust recognizers. These two observations led to my second research question, **RQ2. Can a mobile motion gesture design tool support rapid in situ prototyping of motion gestures with commodity wearable devices?**

5.1 Introduction

One of the major challenges for motion gesture design is time, effort, and expertise needed in building a robust recognizer [114]. This resonates with my personal experience of building a gesture recognition tool capable of performing data acquisition, visualization and classification and supporting user evaluation for a wrist-worn wearable input device. Initially, I sought out commercial tools (e.g., Matlab) or open-source libraries (e.g., scikit-learn [109], Weka [50]) that provide most of the required machine learning functionality. The major holdup with these tools was that they not only required significant and prolonged learning effort but also achieving a streamlined workflow with these disparate tools was challenging and non-trivial. Moreover, my requirements evolved as my understanding of the sensor data and the processing pipeline improved (a challenge also noted by other researchers [108]). In the end, I built a custom solution to meet my needs for the project. Needless to say, building such tools require considerable engineering effort which might not be possible for

Figure 5: Mogeste is a mobile tool for designing gestures (a) with wearable/handheld devices (b) in various naturalistic contexts (c).

the majority of the designer community. Even so, while designing suitable motion gestures the focus of effort should not be on such computationally advanced task but to iterate and evaluate different design ideas.

Another challenge I encountered was around the mobility of the tool. Even though my desktop tool reduced the time for sensor explorations from days to minutes, tethering to a laptop allowed limited testing of interactions outside the lab and limited the mobility of my system. This disconnect between the development and the usage environments [1] could potentially alter design decisions specifically in case of motion gesture design. Thus, I hypothesize, for effective and efficient motion gesture explorations, tools that embody developer's expertise and allow flexibility to explore and iterate would be useful for designers.

This would help mitigate the challenges of building a gesture recognition system and support creative explorations by interaction designers. Furthermore, to address the mobility issue with the contextually rich nature of motion gesture designs, a tool built for mobile platform(s) would allow for explorations across multiple different settings and scenarios.

As a solution, I present Mogeste (Figure 5), a smartphone-based tool, to support rapid, iterative, in situ motion gesture design by interaction designers. It facilitates a create-test-analyze workflow [74] and leverages programming by demonstration [30]. It builds upon previous work in desktop authoring tools for sensor-based interactions by providing an untethered solution to enable design in ad hoc, naturalistic settings. I also present finding from a two-part user study with 7 novice designers which provides an understanding of a novice's design process and also evaluates Mogeste's usability. In doing so, I propose that such a tool would potentially help designers overcome their creative block due to advanced computational needs and make the execution, documentation, and evaluation of motion gesture designs easy and accessible.

In this chapter, I make following contributions.

- Design, implementation, and evaluation of a mobile motion gesture design tool for in situ use by designers.
- Results of a user study with 7 novice designers eliciting their process and emergent strategies.
- Discussion of limitations of my proof-of-concept implementation and new insights for future tool development.

5.2 Design

5.2.1 Why Mobile?

Most rapid prototyping tools in the literature are desktop-based tools [6, 52, 74, 34]. Desktop (and laptop) development provides a large screen, establishing multiple wired and wireless connections, and, more importantly, leveraging mature development and visualization tools. However, size, and operation in proximity limit design in naturalistic and ad hoc settings, particularly for mobile/wearable sensor-based interactions.

Mobile platforms afford portability, boast higher penetration and daily usage, and provide simple direct manipulation interfaces [64]. Moreover, experience with myriad productivity applications shows that, with good UI design, small screens can be used just as easily. Mogeste embraces mobility for in-context gesture design. It also unifies the development and runtime environment.

In hybrid solutions, functions are distributed across mobile and desktop platforms, potentially leveraging the best of both worlds. However, as seen in recent examples [93, 94], in this mode mobile is just used as a sensor platform.

5.2.2 Design Guidelines

As informed by the challenges noted above, the tool should:

- **Enable in-context design.** In-context gesture design refers to the capability to record gesture examples in naturalistic settings, train a recognizer in real-time, and test gestures as commands in an application's context. In this unique way it supports unexpected moments of creativity.

- **Promote a rapid iterative workflow.** Designers should be able to design and assess new gestures in iterative cycles.

- **Support off-the-shelf devices and beyond.** Interaction designers may use commodity devices to quickly prototype form and interactions of an envisioned device. Additionally, custom devices should be equally easy to setup and use.

- **Automate the secondary processes.** By managing data collection, storage, visualization, and networking, designers are able to focus on the core task of design.

5.3 Motion Gesture Prototyping with Mogeste

Leo is an interaction designer who is working on a project to re-design a music player application to be used by users while working out at the gym or running. Since phones are not comfortably accessible, he is seeking to use a smartwatch as a controller. He is exploring motion-gestures as a potential interaction modality, as using touch on the watchscreen with

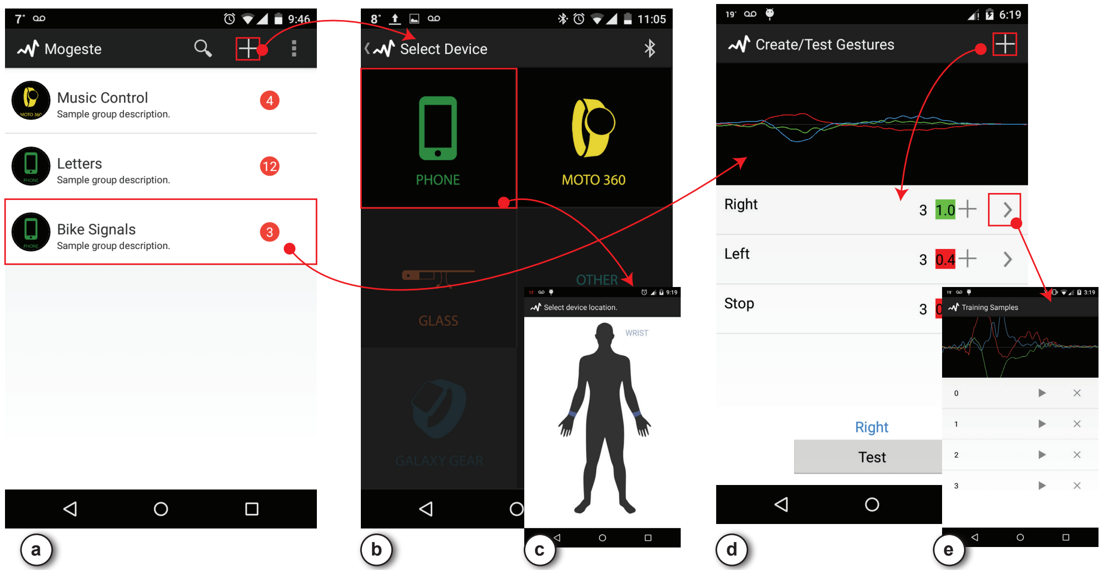


Figure 6: Interface is split in three main screens: a) home screen doubling as gesture group list; b) device selection screen with c) on-body placement selection; d) gesture creation and test screen; and e) sample review screen.

sweaty hands or audio input with other people around might not be appropriate. He brainstorms a couple of gestures and shops around for a good prototyping tool or easy to implement gesture recognizer to make an informed design decision. He settles for Mogeste which offers him the capability to quickly design, test and analyze gestures without any coding requirement or computational overheads.

5.3.1 Gesture Design

Leo opens the app and adds a new gesture group. The next screen prompts him to select a device from a range of supported devices (Figure 6b). Since Leo is designing for smartwatch, he chooses the corresponding icon for "Moto 360". The next screen prompts him to select the body part he wants to design for on a human anatomical model (Figure 6c). Leo wants to test the gesture first with left hand since from his field research he found that people wear watch on the non-dominant hand. Because majority population is right-handed, he wants to first design gesture for right-handed users. Leo renames the group to Music player. The meta information for the group has details about the device and body part for which the gestures (referred to as gesture class(es)) are being designed and also number of gestures inside the group (Figure 6a). Leo can put all the left-handed gestures for the music player inside one group.

Once inside, Leo adds a gesture class to the group (Figure 6d). The meta information for the class has details about the confidence value (discussed later) and number of samples within each class. To allow recording of gestures with the watch, he first starts a helper application on the watch which transmits sensor data to the paired smartphone where gestures are stored. He then renames the gesture class to 'Next' and provides a sample to the system. He translates his wrist in the right-direction. Mogeste is designed to provide a 2 second time interval for the user to record the sample and the elapsed time is shown through a circular progress bar. Once the recording is finished, a visual representation in the form of line chart gives live feedback of the incoming sensor data for the recorded gesture. He provides more examples for the gesture and moves on to record another gesture for 'Previous'. Mogeste also provides him with the option to review samples and delete the undesirable ones (Figure 6e).

5.3.2 Gesture Testing

Once Leo records all the gesture classes, he tests them if they are being recognized correctly or not. He hits "Test" button (Figure 6d bottom) and does the intended gesture. There is an audio and textual feedback for the detected gesture. Leo is excited because the application correctly detected his intended gesture.

5.3.3 Gesture Analysis

Leo notices a goodness value for the gesture classes he designed which ranges from 0-100% and is color coded (red for low score and green for higher). Higher values denote uniqueness of a class.

Mogeste presents the designer with an inter-class gesture "goodness" score [6], whenever a new motion is performed. A class' goodness is based on the F-score (a combined metric representing precision and recall) using 10-fold cross-validation of the class compared to every other class in a gesture group.

Leo decides to record multiple samples while running, biking and walking and check what gestures are reliably detected. Mogeste provides him with a simple and easy to use interface to quickly prototype and test gestures in situ.

5.4 Implementation Details

5.4.1 Hardware

Mogeste is written using the Android SDK v5.0. I tested it on a Nexus 4 smartphone that connects to a Moto360 smart watch through the Android Wear API, and a Google Glass (2nd edition) eyewear through the Glass Development Kit (GDK). Each of the three devices has a 3-axis accelerometer and a 3-axis gyroscope, operating at the highest sampling rates of 200 Hz, 100 Hz, and 30 Hz, respectively. Inter-device communication is via Bluetooth and connections are managed automatically. Data is stored in the JSON format on a database on the phone.

5.4.2 Software

Mogeste uses a Sequential Minimal Optimization (SMO) implementation of Support Vector Machines (SVM) for multi-class classification and the LibSVM [24] implementation for unary classification from Weka [50] for Android[1]. SVM requires minimal parameter tuning while still performing well with limited training samples and a high number of features. Moreover, analysis performed by both Bulling et al. [16] and Wu et al. [135] positions SVM ahead of commonly used alternatives found in the literature, such as Dynamic Time Warping (DTW) [42] and Hidden Markov Models (HMM) [110]. I extract the following features per sensor channel: root mean square (RMS), standard deviation (SD), mean, and normalized sensor energy.

5.5 First-Use Study with Novices

In this section, I describe a summative study conducted with 7 participants using an early version of the tool. The study was conducted in two parts with the goal of understanding a novice's design considerations and evaluating the tool.

[1]https://github.com/rjmarsan/Weka-for-Android

5.5.1 Participants

I recruited 7 second year graduate students (five female, two male) from our HCI program. No participant had prior experience with designing motion gestures but was familiar with their availability on commodity wearable/handheld devices. Therefore, I consider them novice designers. With the exception of one participant, none of them was comfortable coding and had not been exposed to pattern recognition.

5.5.2 Procedure & Tasks

I had two parts to my study: design exercise and tool use. Each of these was followed by a semi-structured interview. The study lasted for an average of one hour. The goal for the design exercise was to establish prior knowledge, while tool use was an evaluative study to get user feedback. To seed the discussion, participants were introduced to motion gestures. Then they were asked to come up with three alternative gesture designs for two given tasks of *placing a call* and *ending a call*. The participants were asked to assume that all the preceding steps like selecting the contact have already been done. They followed a think aloud protocol and were provided with a phone to bodystorm. They were provided materials to write or draw their designs on a worksheet. The follow-up interview focused on understanding their design rationale and how they would prototype and evaluate their designs.

For the second part of the study, the participants were given a brief demo of various features and workflow of the tool. Subsequently, they were asked to prototype their six gesture designs using Mogeste while following a think aloud protocol. The interviewer answered any queries they had regarding the tool. The participants were given freedom to complete the task while being stationary or mobile but within the confines of the field of view of the camera. Also, I let them choose a comfortable pace for themselves. Post tool use, another semi-structured inquiry was done for gathering feedback about the tool usage and its features. I specifically asked questions about their experience, likes, dislikes, and suggestions for the tool.

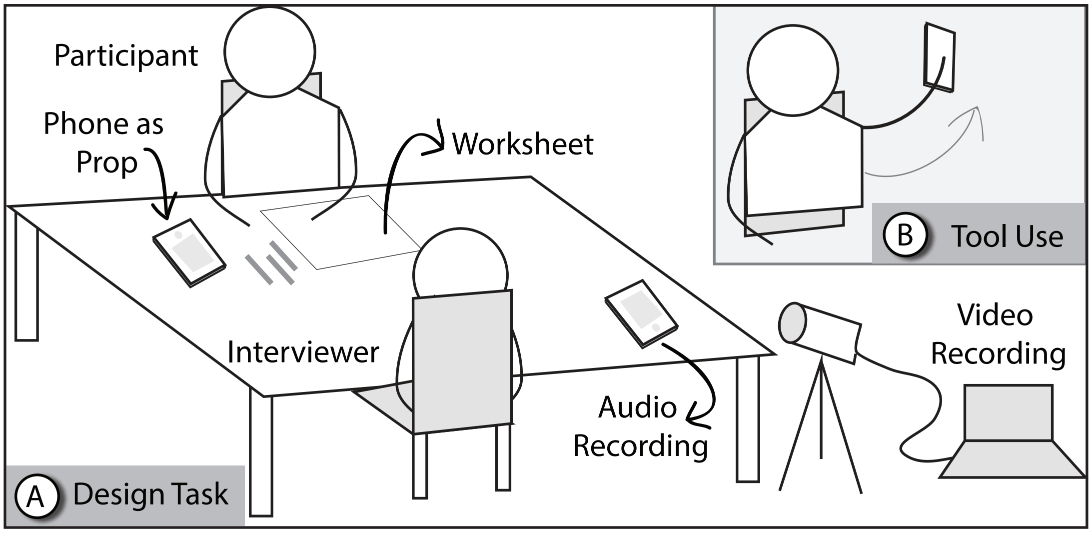


Figure 7: Mogeste first-use study setup.

5.5.3 Setup

Figure 7 shows my setup. Participants used Mogeste installed on a LG Nexus 4 smartphone running Android 5.0.1 to implement their designs. All sessions were video and audio recorded for which I used a web camera connected to a laptop and another smart phone respectively. I also recorded the screen of the device on which the application was running using an existing screencast application.

5.6 Study Results

Here, I present the findings from my two-part study. First, I discuss the insights from the design exercise and, then, move on to discuss the findings from the evaluation study.

5.6.1 Design Exercise Feedback

During initial design exercise, participants brainstormed a few ideas and while relating to their everyday experiences used terms such as ‘intuitive’, ‘natural’ to describe their designs. As intended, they used the phone as a prop while enacting their gestures. Some chose to textually describe their gesture designs, whereas others drew sketches. Most of the participants came up with gestures that either mimicked natural actions (e.g., phone to ear) or were simple (e.g., shake, draw in air), confirming Ruiz et al.’s [116] findings. However, in a few exceptional cases, participants came up with more complex gesture designs. For example, one participant devised compound gestures involving ordered sequence of simple gestures to avoid the accidental trigger. Another participant did the same when they realized that their original idea of shake gesture would fail because they tend to fiddle with their smartphone. Another participant suggested a three-second duration for their gesture with repetitive motions to indicate an intentional action. Lastly, one participant employed repetition of the same gesture twice as a strategy for avoiding false triggers.

During the follow-up interviews, when asked about how they would go about prototyping their gesture designs, the participants either confessed their lack of expertise in programming, or suggested low-fidelity prototyping techniques such as sketching or animation. Although they had a sense of what information was being sensed, they weren't confident in implementing a recognizer that leveraged it. In response to specific questions about who would they collaborate with, I saw a clear dependence on developer's expertise in all the responses. For example, one participant suggested she will come up with ideas and help with conducting user studies but will want a developer to implement gesture recognizer for her. This evidence confirms my hypothesis and the need for a prototyping tool which would allow designers to explore irrespective of their coding skills. Upon asking "what would an ideal tool look like that could help them through the design process?", participants indicated towards a tool that would help them 1) record gestures in different situations, 2) set thresholds/range to classify motions as gestures, 3) evaluate and compare gestures, and 4) provide cross-platform support. Although the pool of participants was small, I believe there is a qualified need for an easy to use prototyping tool for designers interested in prototyping motion-based gestures.

5.6.2 Feedback from Tool Evaluation

All the participants were able to complete the given tasks and had a positive response to the tool and remarked that it was "cool". I also observed excitement during tool use every time the recognizer produced a correct label corresponding to a test sample. Some participants also elaborated that the tool was accurate. Furthermore, most participants considered the tool to be powerful yet easy to learn. They attributed its power to absence of coding, time saving, and refocusing efforts to the core exercise of design.

They highlighted the tool's potential utility for the entire design process, from early stage exploration to user testing. For the early stage brainstorming, the tool led to more ideas. Also, the participants said that the tool helped evaluate the feasibility and was good for initial validation, that is, to determine whether it was "worthwhile going further with the design". For the prototyping phase that follows brainstorming, the participants found it easy and quick (few seconds) to record gestures by demonstration. Finally, thinking about the evaluation of designed gestures, participants noted the tool would be useful for deployment and would facilitate rapid testing with other people. Ability to make different groups of gestures was found relevant in this context. As it is, they would use the tool for measuring the performance of a recognizer. As their confidence in the tool increased, a feeling of empowerment emerged. For instance, a participant could imagine using the tool to "show the developer that this gesture is doable." Another participant said "I wouldn't have been able to do activity recognition myself. It took care of all that."

In addition to the utility of the tool to motion gesture design, participants used it as a means of reflection and analysis. One participant stated that the tool "helps understand that some gestures are easy and some are difficult" to perform. Another participant noted that Mogeste "actually turns it (gesture designs) into something tangible." I also observed that the participants paid close attention to the confidence/goodness score and used this real time feedback to add and remove recorded samples and on some occasions changed their designs altogether. In future, I wish to investigate recognizer feedback as a means of reflection further.

While most participants used the tool while sitting at the desk, one participant picked up the phone and started strolling while enacting gestures. Most other participants freely moved the phone while sitting without restrictions (Figure 8). I expect to see more of this when conducting studies outside the lab setting in future.

In order to test limits of my tool, I chose not to delete any participants data from the tool during the study. As a result, towards the end it had 10 gesture groups with number of

Figure 8: Participants moving freely with the phone-based tool. (Right) Participant stands and walks to help with ideation.

classes within them varying from 1 to 7 and up to 7 samples for each class. Note that this is indicative of long term use of Mogeste. As more data was accumulated, Mogeste took slightly longer to fetch data from database and populate a page, but otherwise it performed as expected.

5.6.2.1 Opportunities for improving the tool

Although the tool was functional, I discovered some limitations and new opportunities based on the feedback I received from my participants.

My biggest insight was about implied constraints on start and end position of gestures due to dependence on visual feedback. For example, the most common gesture that participants came up with for ending the call required them to start gesture recording by tapping on the screen when the phone was next to their ear. This caused them inconvenience as they had to tilt their heads at acute angles to look at the screen (Figure 9). If the participant instead chooses to start recording by first bringing the phone in front of their eyes and then quickly moved it to the intended start position (next to ear), the recorded gesture would also include this extra movement. Another related issue was a lack of non-visual feedback when the recording started and ended. Owing to this issue participants sometimes wondered if the recording had started or when did it stop. I believe the inclusion of speech output delimiting the gesture recording along with speech input to initiate recording will significantly improve user's experience.

Figure 9: Participants discovered inability to see visual feedback when the gesture start and end point were in inconvenient locations.

Another concern was related to fixed length (2 second) of the gesture. A participant reflected, "if gesture is complicated you need more time to perform it. If the gestures is super simple then it's too long." Perhaps relaxing this constraint by either making it flexible, similar to MAGIC [6] or allowing it to be customized per gesture class will address these concerns

Despite the use of an anatomical representation of human body for selection of body placement of a device (see Figure 6c), there was a confusion about left and right direction from participant perspective and device perspective. Adding labels would alleviate this problem.

My tool provided one type of recognizer feedback via confidence score, but without any recommendations for how to act on that score. When it was low, a participant asked "should I provide more examples?". Conversely, when it was high, she asked "when should I stop recording more examples?". As a means of guiding designers when recognizer feedback leads to confusion, Long et al. propose incorporation of contextual, actionable design advice [89].

All the participants declared Mogeste's line-chart visualization of incoming sensor data "hard to read" and use, except as a confirmation of changing values. Because 3-dimensional inertial sensor data is not as easy to visualize as 2D traces on a touchscreen, further investigation is needed. On a related note, my participants suggested the addition of a visualization for showing intra-class comparisons as well as a flowchart for denoting progress through tasks of a project. More specifically for indicating progress, one participant wanted to browse the test logs to review how many times certain gesture was rightfully detected or what gesture was it most confused with.

Regarding recorded samples, my participants suggested starting sample numbers (see Figure 6e) from 1 and using timestamp or commenting capability so that they can recall what sample was recorded when they access it in future.

The test version of Mogeste supported inter-class comparisons within a group and multi-class classification. Participants suggested allowing to test an individual gesture class and also comparing gestures from different gesture groups. They also wanted to be able to view how different is the test sample from the training samples. Yet another feature request was to be able to tell system that it has recognized a wrong gesture and provide a better sample.

Thinking from a developer's perspective, a participant who was familiar with coding requested the ability to export the recognizer and sample code to bootstrap its implementation into an application.

Table 5: Participants employed two strategies for selecting gestures for opposite but co-occuring tasks: *same* gesture, and *opposite* gestures. I list example labels from the study to illustrate this.

Tasks	*Same* gesture	*Opposite* gestures
Place call	rotate call	pull out
End call	rotate end call	put back end call

Furthermore, I identified two strategies that designers employed when creating gestures for tasks that normally occur in pairs, for example, place and end call. First, if the tasks always occur in the same order then the same gesture can be utilized for both actions without any ambiguity (Table 5). Alternatively, because the tasks have opposite meaning, opposite movements can be associated with each task. In retrospect, both strategies seem natural and useful for reducing need for inventing new gestures. I hope to better support such emergent strategies within my tool.

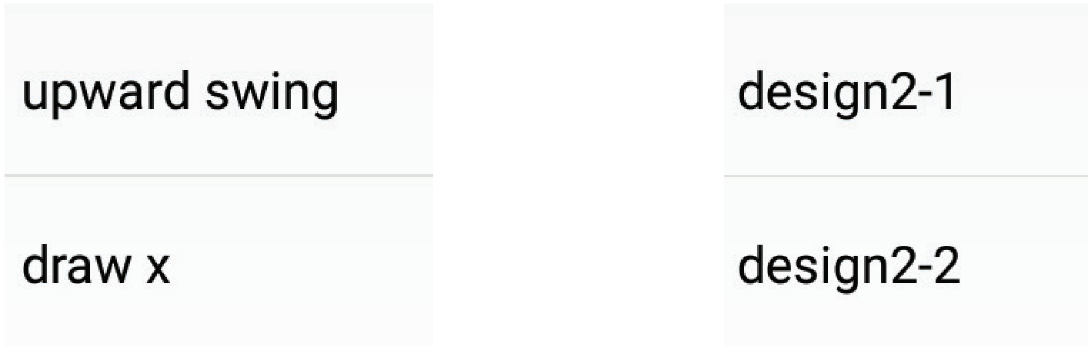


Figure 10: Some participants choose semantic labels (Left) whereas others used numeric labels (Right) for different classes.

I also observed that the participants used different naming strategies for class and group labels. While some participants used numeric suffix labels to denote different designs corresponding to the same task (Figure 10), others used semantic labels referring to the performance of the gesture (Figure 10). In absence of other means to aid recall, my tool should have enforced a naming convention hinting at the meaning or actual performance of gesture. However, since 3-dimensional motions are not easy to explain, this strategy might backfire too. For instance, the description might become too long (see Figure 11). Perhaps enabling optional longer descriptions separate than the label would be more appropriate.

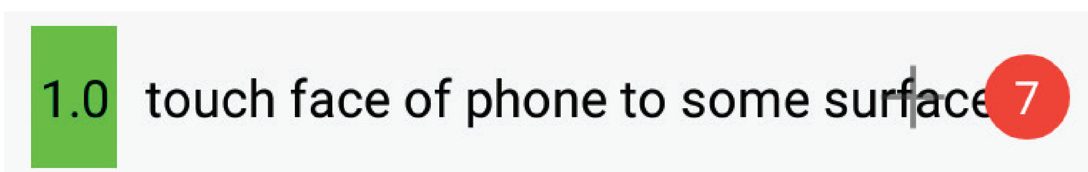


Figure 11: A long descriptive label disrupted the UI.

As for the group labels, participants followed similar strategies (Figure 12). As seen in the figure, participant p4 choose to use the default group label "New Gesture" that my tool assigned. I noticed that this could be because workflow within my tool took the participant directly to the "Create Gestures" screen (Figure 6d) without allowing them to label the gesture group first. I believe that this workflow also led participants to create all gestures within the same group. A minor tweak that allows complete specification of the group before creating classes within it would help prioritize better naming of groups.

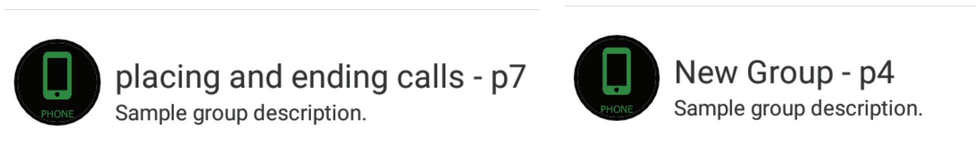


Figure 12: Participants used different naming strategies for gesture groups. Note that I added participant IDs p* afterwards.

In the absence of feedback while the system was loading data and populating pages in the background, occasionally participants became impatient and started tapping the screen furiously, which led to few crashes and unintentional recording of samples. However, the tool recovered to a normal state upon restarting the application.

5.7 Implications for Design

From the user study, I realized that gesture design and gesture prototyping are not independent of each other but are co-dependent and co-occur. By allowing a designer to try different designs with little effort (few examples per gesture), a gesture design tool can help her focus on deeper questions about what's possible, what makes sense to the task and end-users, how can she evaluate her designs, etc. A tool can also help by making documentation of the design process easy. This is particularly important during early-stage prototyping when she is working on multiple design alternatives, therefore, recollection of details of each is poor. Moreover, she may want to quickly annotate designs with observations made during an in-context evaluation. By offloading processes around the design task, a prototyping tool can help refocus resources at the gesture design and provide a seamless platform to manage different tasks and activities.

A designer considers gestures as a means to an end, therefore, the context of use or application is inseparable from the gesture itself. Moreover, they want to evaluate their gesture designs within that context. Perhaps, a facility to link gestures to application controls will be useful. Considering the physical characteristics of the gesture performer as another piece of context, one study participant wanted to record who performed it. Her rationale was that "children might perform gesture like this but adult will perform gesture like that, maybe its a little diff(erent)."

5.8 Placing Mogeste in the Design Space

Based on the dimensions highlighted in Chapter 4, I have implemented Mogeste—an initial prototype of a mobile tool for in situ motion gesture design built using a modular framework (see Figure 6). My tool targets interaction designers and supports the prototyping and evaluation phases of the designer's workflow. Designers are able to create and test motion gestures by demonstration, using inertial sensors, a touch-based mobile interface, and a black-box machine learning back-end. Currently, my tool follows a many-to-many sensors-to-gestures approach, using both accelerometers and gyroscope to design discrete atomic gestures. Mogeste abstracts background tasks for data collection, storage, and visualization of discrete gestures. Designers are presented with a static line chart representation of the raw sensor data after the gesture is performed and captured. After training data has been collected by demonstration, the tool allows the designer to test a new gesture, within the tool and without connections to the applications, and outputs the predicted classification. In addition to the visual and auditory feedback of the recognizer's classification, Mogeste also provides the designer with a uniqueness value based on an inter-class comparison. Mogeste embraces mobility for in-context gesture design, unifies the development and runtime environments, and facilitates conducting participatory and laboratory studies (.

5.9 New Features in Mogeste

Based on feedback from the first-use study I propose several improvements to Mogeste tool that enhance a user's experience. First, I enhanced sensor data collection procedure by adding a countdown, audio feedback, and speech-based triggering mechanism. Second, I added support for video recording and playback of performed gestures. Last, I am working on a tweak to the testing functionality to allow a user to confirm predictions of a recognizer and, then, automatically add labeled test samples to the training data to improve future predictions.

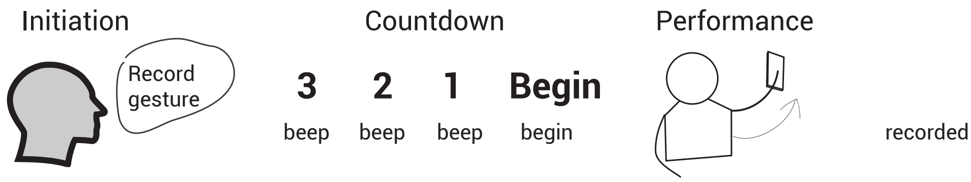


Figure 13: Enhanced procedure for collecting gesture examples.

5.9.1 Enhancements to Sensor Data Collection Procedure

While prototyping gestures with Mogeste my study participants expressed concerns regarding inability to initiate gesture recording from a starting position where the screen is not visible. Additionally, they could not ascertain whether the recording is complete or ongoing without looking at the screen. Both of these issues underscore the need for a gesture example recording procedure that can be initiated and monitored without having to look at the screen.

Therefore, as shown in Figure 13, gesture recording can now be initiated by speech also. Upon initiation a countdown runs for 3 sec with both visual and audio feedback, and ends with an announcement to 'Begin' recording. After the gesture is performed or the prescribed time limit is reached it announces 'Recorded', marking end of recording. To record more examples, the designer can repeat the procedure as many times as she wants.

A clear advantage of this feature is an improvement in consistency of recorded examples. Another advantage is hands-free triggering of recording, which is particularly useful when emulating a use case where hands are occupied.

5.9.2 Video-based Recollection & Communication

Mogeste and several other authoring tools for gestural interaction provide only linechart visualizations of the recorded sensor-data stream. However, this representation is neither comprehensible by designers nor it is helpful in recollection of the performance of the gesture. Moreover, it also limits communication of gestures to end-users and team members unless the designer is present in person. To overcome this limitation, MAGIC [6] allowed video recording of gestures from a head mounted camera pointing downwards and also supported playback of videos associated with gestures. Other researchers [67] have also confirmed the usefulness of videos for teaching motion gestures. Unsurprisingly, Android smartwatches come with video-like animations as tutorials for unintuitive gestures.

Due to the compelling evidence in support of video-based communication, I too investigated the feasibility of automated recording of video during gesture performance. To enable this feature I made a simple assumption that professional designers can afford an extra device for just recording the video. However, unlike MAGIC where video recording is done through a single-purpose web camera attached to a desktop, I utilize the same platform for interface design and video recording, namely a smart phone.

Let's see how the video recording works. On the gesture training screen, live camera preview is shown on top one-third. Although by default the rear camera is active, here the designer can also switch to front camera. Live preview helps designer rehearse gesture performance and make sure the full range of motion is within the camera's view. Now when the designer initiates recording of example as explained before video and sensor data are simultaneously recorded. Once recorded each video is automatically attached to its corresponding gesture example and can be played back in the gesture example screen. The collection of videos for a single gesture can reveal any unconscious adaptations a designer made to the performance of a gesture. Although having a video per gesture example is useful during prototyping, ultimately a designer would choose one canonical video per gesture and maybe discard other videos. I currently do not allow deletion of videos.

With speech input and video recording on phone designer can explore both first-person and third-person perspectives. For first-person I am assuming the phone is hanging from the neck on a lanyard, similar to Gesture Pendant [126] and SixthSense [96]. For the Both perspectives have different affordances and limitations. First-person point-of-view (POV) is more appropriate for recollection, particularly useful during mobility, and can lead to more consistent results. However, a narrow field of view prohibits recording of gestures that occur lower than abdomen or above neck height unless the camera is manually adjusted. Additionally, privacy concerns may arise when passersby are recorded without permission. But several filtering techniques could be explored to mitigate privacy concerns.

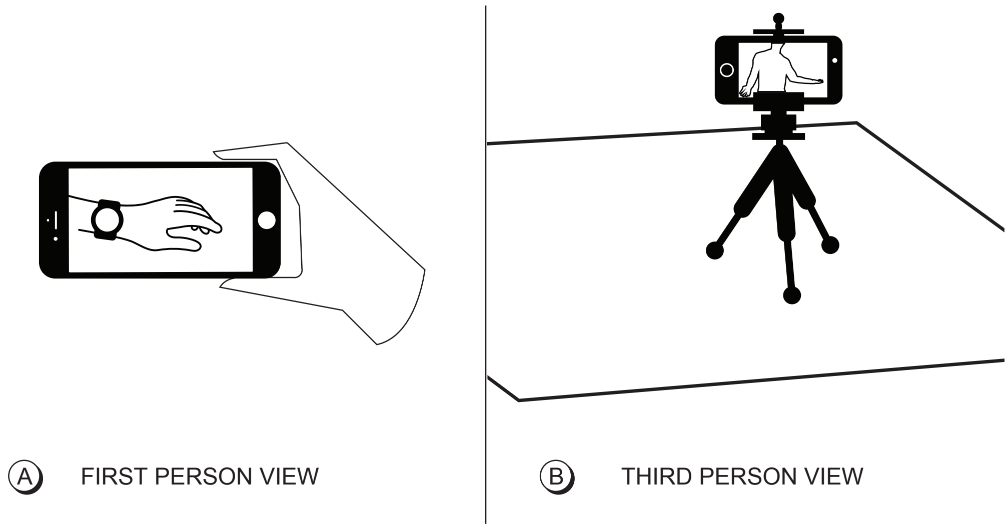


Figure 14: Showing first-person and third-person configurations.

An alternative solution is to capture third-person POV using the front camera. In this case, the camera is typically steadily mounted on a tripod or stand placed at a distance from the body. This configuration frees the designer to perform any kinds of gestures in a much broader field of view than first-person POV scenario. Note that recording can only be initiated through speech input. The biggest drawback of this setup is the lack of mobility. Another concern is its suitability for allowing replication of exact gesture by others with whom video is shared. Nevertheless, third-person POV opens up an opportunity to utilize vision-based tracking in addition to inertial sensing for better accuracy.

5.9.3 Each Test Example is a Training Example too

Traditional machine learning prescribes recording of a large amount of data which is split into training and test data, conducting offline processing and analysis, exporting a recognizer based on training data and then testing it offline on test data. Sometimes when online testing is performed the exported model is run live on incoming streams of sensor data, however the test data is not recorded and used for improving the model. Moreover, large training set is almost always a prerequisite.

On the other hand, interactive machine learning (IML) assumes very little training and expects to iteratively improve the recognizer with human input. For example, in Fails and

Olsen's Crayons system a developer helps improve a vision-based motion tracking system by marking regions of an image that are human skin [39]. With Mogeste, I am enabling rapid early-stage exploration of motion gestures through prototyping. Hence, I can't expect a designer to spend too much time on training one of the several gesture candidates, which means they can at best provide a few examples of each gesture. In many of the user study sessions, I have found an average of five gesture examples to be reasonable. However, since I always observed a test session following a training session, I wondered if the tool can allow a designer to correct its predictions and then include the test examples as additional training data for next round of testing. In this simple way, Mogeste can generate a better recognizer every time it is tested with a human's help.

5.10 Summary

Mogeste's mobility, simple UI, and novel features enable designers to create diverse gestures in complex naturalistic settings through simple demonstration, fulfilling the criteria set forth by Hudson and Mankoff [63] for effective tools. Furthermore, through Olsen's situation-task-user (STU) framework [104], I characterize Mogeste as extending the task of motion gesture design to mobile situations (e.g., sitting, running) for interaction designers who would otherwise be hesitant to participate due to their lack of expertise in programming and pattern recognition. Finally, Mogeste refocuses the designers' time and effort on gesture design by leveraging programming by demonstration and promoting an iterative workflow. From an evaluation with novice designers, I validated my hypothesis that such a tool will empower designers. Furthermore, I identified specific areas for future developments in mobile motion gesture design tools.

CHAPTER 6

COMOGE: CONTEXT-BASED PROTOTYPING OF MOTION GESTURES IN SITU

When motion gestures are designed out of context, designers miss important contextual cues that can cause well designed gestural interfaces and best performing recognizers to fail during real world use. Therefore, I propose an *in situ* gesture prototyping approach which leverages contextual information. This leads to my third research question, **RQ3. What impact does context have on a gesture's design and a designer's gesture prototyping process?** In this chapter, I present *Comoge*, a standalone mobile tool for designing gestural interfaces, and my evaluation of this tool. Comoge facilitates capture and use of six types of contextual information: 1) activities during which gestural interactions are performed; 2) location and time of training/testing; 3) form factor of the sensing device; 4) placement of the wearable device on the body; 5) remarks about social context and perceptions; and 6) application genre and intent of interaction. Findings indicate that Comoge makes it easy for designers, regardless of their expertise, to rapidly prototype gestures in context while using contextual cues to inform their gesture designs.

6.1 Introduction

"It depends on the context" is as true for designing usable, contextually appropriate, and machine-recognizable motion gestures as it is for understanding my everyday actions and decisions. 3D motion gestures are readily sensed by accelerometers and gyroscopes already embedded in most handheld (e.g., smartphones) and body-worn (e.g., smartwatches, Google Glass) devices nowadays. While motion gestures are found particularly useful in situations where visual attention is limited [101], or when brief interactions with wearable devices are needed [7], with personal computing devices that accompany us everywhere, opportunities for gestural interactions abound.

But when motion gesture recognizers are developed out of context using desktop-centric development tools [52, 6], designers often miss important cues associated with the context in which the final application use case is situated. Consequently, gesture recognizers that work well in controlled environments, often perform sub-optimally in naturalistic environments. I strongly believe that just like freeform motion gestures "are likely to help users think and communicate"[73], tools that free designers from the confines of a laboratory, can unleash their creativity through in-the-moment iteration and reflection support. Initial mobile tool research [106, 68] shows feasibility of *in situ* gesture prototyping. However, without an explicit support for *in situ* capture and analysis of the information pertaining to the contextual factors that affect a gesture's acceptance and viability, designers will neither be able to prioritize usage scenarios nor will they be equipped to identify causes for errors.

Therefore, I developed *Comoge*, a standalone smartphone-based tool that enables any interaction designer to conduct rapid, *in situ* design and evaluation of motion gestures performed with various wearable and handheld sensing devices. Through a simple user interface, it also facilitates the human-assisted capture, access, and use of the contextual information that impacts usability, appropriateness, and recognizability of a gesture. It currently supports six types of contextual factors: 1) activities during which gestural interactions are performed; 2) location and time of training/testing; 3) form factor of the sensing device; 4) placement of the wearable device on the body; 5) remarks about social context and perceptions; and 6) application genre and intent of interaction. In this chapter, I describe the design criteria underlying Comoge's development, its design, implementation details, and its evaluation with professional and aspiring (student) interaction designers.

I learned that Comoge provided the study participants, especially the ones who had never designed gestures before, with a reference for an in situ gesture design process that encouraged experimentation and rapid iterations. On the other hand, more experienced participants saw utility in integrating Comoge into an existing workflow to save them time and effort. Furthermore, Comoge's easy to learn and use interface made designers cognizant of differences between contextual settings. Though most of the participants were fearless when it came to performing gestures in public spaces, they showed awareness of onlooker's perceptions, found ways to manage and manipulate those perceptions, and considered their own judgment of a gesture's social acceptability to be incomplete without a third-person perspective. As a direct evidence of the impact of contextual information on a gesture's design, a few participants changed their designs after testing while most others had ideas for what they would have changed.

This chapter makes three main contributions. First, I identify and describe contextual factors relevant to the design of motion gestures. Second, I provide implementation details and demonstration of *Comoge*, a smartphone-based tool that enables interaction designers to design, prototype, and evaluate motion gestures in the desired usage context while allowing easy capture and access of contextual information. Finally, I share results of and insights from a summative evaluation conducted with 5 professional and 12 aspiring (HCI graduate students) interaction designers.

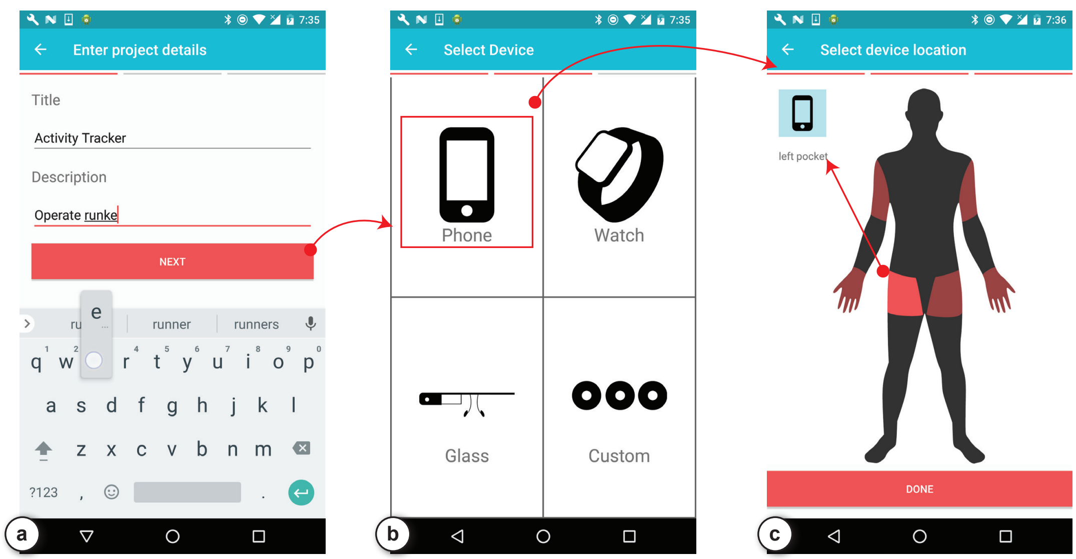


Figure 15: Comoge gesture/project creation wizard: (a) captures title and description; (b) allows sensing device selection; and (c) lets the user choose body placement(s) of the selected device.

6.2 What Constitutes Context for Gestures?

Dey [32] defined context as "any information that can be used to characterize the situation of an entity. An entity is a person, place, or object that is considered relevant to the interaction between a user and an application, including the user and applications themselves." I consider following six types of contextual cues critical to the task of prototyping gesture recognizers. I provide below a description of each of the six contextual dimensions, mode of capture, and discuss their significance. A discussion of their visual representation is addressed in the next section.

Activities during which gestural interactions are performed: End-users expect the same set of gestures to work irrespective of the activity they are involved in. Hence, designers should either select gestures that are activity agnostic or train and test gesture recognizers with representative samples from each activity context of interest. However, both approaches have severe limitations. Activity agnostic gestures are very difficult to find, and training a single recognizer with a wide variety of data can lead to poor performance due to low inter-class variance. A potential solution is to train multiple recognizers for the same gesture, one per context (for example, "Wave *while running*" and "Wave *while biking*" are two different recognizers) and while testing employ only the recognizer relevant to the current activity context. Tagging each gesture sample with activity context makes this process fast.

I acknowledge that there is a wider space of activity contexts, but for the purpose of this chapter, I focus on a categorization based on mobility because being in motion directly affects the sensor data. Comoge currently supports gesture design while being *still* (standing, sitting, or lying down), *on foot* (walking or running), and *in vehicle* (biking, driving, flying, or riding a bus/train).

Location and time of training/testing: Collecting gesture samples in controlled environments is very different than recording them in naturalistic environments. The same natural environment may provide different constraints during different times of the day. For example, occupancy inside a workplace may vary throughout the day. Location can be *physical* (GPS coordinates) or *symbolic* (such as work, home) [53]. Physical location is easier to calculate, but the symbolic location is more useful for this work. A designer may consider a broad classification such as indoor versus outdoor, or one that relies on specifics such as work, home, gym, mall, etc. Because fully automated determination of symbolic location labels is still not possible, perhaps intermittently I can ask designers to manually tag each sample or may infer it from the activity context.

Form factor of the sensing device: Based on the use case, a designer chooses a form factor for the sensing device. When envisioning new interactions for an existing form factor they typically choose a commodity platform such as an Android smartphone. Conversely, when prototyping gestural interactions for a new gadget they might use a commodity device as a proxy or choose to work with custom-built hardware. A designer can input this information before starting the gesture prototyping activity. Closely related to the device are available sensors. Sensors chose for prototyping limit the types of motions and the resolution that can be detected. Often a commodity platform discloses information regarding available sensors through an API, but when it doesn't, a designer can input this information manually. With Comoge, a variety of custom and commercial form factors can be used, but the sensors are fixed to accelerometers and gyroscopes.

Placement of the wearable device on the body: Once a sensing device and its sensors have been selected, a designer will need to specify the placement of the sensing device on the body. This information can largely be inferred for commodity devices but is an open question for the design of a custom device. Moreover, some commodity devices can be mounted in multiple locations with straps and mounts, therefore disambiguation is needed. For example, a phone can be held in either one of the two hands, or strapped to an arm or hung from the neck through a lanyard. Along with device type, the on-body location of a sensor is used for grouping gestures together and disambiguate the recognizer to be used while testing gestures. In relation to placement, a designer should be cognizant of the impact of size or proportions of individual bodies.

Remarks about social context and perceptions: Performance of a gesture in any social situation has implications. For example, repetitive performance of awkward motions with any body part can raise eyebrows or draw unwanted attention. Additionally, gestures can be misinterpreted depending on the cultural context. Many of the nuances of social interactions can only be observed by a designer in a naturalistic context and should be considered in the design of gestures. However, if not captured in-the-moment these

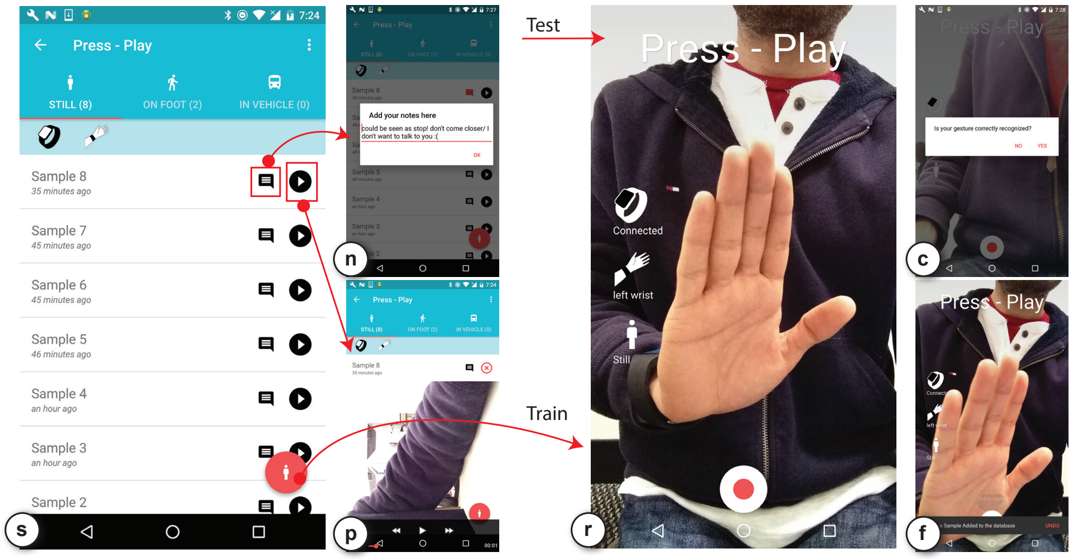


Figure 16: (s) List of recorded gesture samples per activity context (sorted into tabs) with timestamps for when they were recorded. (n) Clicking callout button allows entering notes about social or cultural observation. (p) Video preview of the recorded sample shown in place. (r) The screen where sensor data and video is recorded while training or testing. Notice the different iconography showing different contextual information. (c) The user corrects or confirms test prediction label. (f) Gesture recording finished and undo allowed.

important observations can quickly fade from memory. Two ways I enable the capture of social and cultural contexts are a video recording of a gesture's live performance and textual annotations entered for each gesture performance (or the gesture as a whole).

Application context is determined by both the application genre (such as a game, or a productivity app) and the intent of interaction within an application. In the case of application genre, when a user opens, for example, a gaming application on her smartphone it sets certain expectations about the interaction behavior which are different than if she was working on an email client. However, even within a gaming application, when the user is on the settings page instead of the gaming environment, the intent of configuring settings may influence the selection and performance of gestures. Belotti et al. [11] term this categorization of interaction behavior based on application type and state as *genres of interaction*. Comoge currently does not infer application context, instead, it allows users to define projects, which can be seen as a simple embodiment of the genre/intent they have in mind.

6.3 Motion Gesture Design with Comoge

Comoge is a standalone smartphone-based tool that enables a designer to conduct in situ design and evaluation of a variety of on-body motion gestures performed with various wearable and handheld sensing devices (both custom and commodity hardware is supported). It facilitates a create-test-analyze workflow [74] and leverages programming by demonstration [30]. Comoge extends previous work in gesture authoring tools by providing a mobile, untethered solution that captures and leverages contextual information. For the factors that affect usability, recognizability, and appropriateness of a gesture, it facilitates the capture, access, and use of the contextual information through the following flows.

6.3.1 Gesture & Project Creation

Let us assume that an interaction designer, Jane, wants to design a gesture that controls the "play" functionality of a music player. To achieve her goals, Jane would first open the Comoge app on her Android smartphone which shows two tabs, one each for projects and gestures. Projects can be understood as a container for gesture explorations related to an application scenario. She would then proceed to create a gesture within a project titled "Music Player" or by switching to the gesture tab and clicking on the "+" button. In the latter, she would be presented with a screen (Figure 5a) to enter the name and description of the gesture. On entering that information and pressing the "NEXT" button, she is presented with a screen (Figure 5b) to select the device with which the gesture is going to be performed. She selects the device, thereby also selecting the sensors of that device for data collection, and then presses "NEXT" again. Now she is presented with a screen (Figure 5c) to select the body part where the previously selected device will be placed while performing the gesture. She selects the body part by selecting the location on the outline of a human body presented to her and then finally presses "DONE". In following this wizard, Jane has provided Comoge with the following contextual information, 1) Device being used and 2) Body Location of the device. The information collected here is presented to Jane in all future steps through icons and labels. Jane is then taken back to the list of gestures she has prototyped with Comoge so far. Project creation also happens through a similar wizard. Any gesture created within a project automatically inherits device and body location information. Gestures can be easily added to or removed from a project.

6.3.2 Gesture Training & Testing

To record samples for training purposes, Jane then opens up the gesture by selecting it from the list. She is presented with an empty screen (Figure 16s) with three tabs, where each tab is a container for its respective activity context. Here the contextual information already collected is again highlighted through icons and each tab shows a total number of samples for their respective context. To add samples to a particular context, Jane first switches to that tab and then selects the circular button on the bottom right. By following this process, Jane has again provided Comoge with the information about the activity context in which a sample will be recorded. Next Jane is presented with a screen (Figure 16r) with a live preview of the camera. The screen also presents information about the gesture such as name, device, body location through labels and iconography. After opening this screen, Jane also opens up the companion app on the device with which the gesture will be performed. Opening the app on the device automatically triggers a connection between the devices running the two apps (Comoge and the companion app). To record a sample, Jane then presses the record button, gets ready during the countdown, performs the gesture with the device and then presses the record button again to stop the recording of the sample. At this time I also automatically record contextual information such as time and location in the background. The sensor data received over Bluetooth is saved as a sample with other information about the gesture and Jane is notified that the sample was saved (Figure 16f). If she made a mistake, she can easily undo recording of the last sample (Figure 16f bottom right). Now Jane can continue to record more samples in a similar fashion without leaving the screen or she can go back to the list of gestures within a project to perform some tests.

Gestures within a project are tested together. Jane records a test sample on a screen very much like the training screen. The only difference is that after collecting the sample Comoge presents Jane with a recognition result and a dialog (Figure 16c) to confirm or correct the result.

6.3.3 Gesture Review & Recording Observations

Jane can preview each sample by going back to the sample list screen (Figure 16s) and pressing the "play" button on the respective sample to open up the video (Figure 16p). Lastly, each sample has all contextual information except for information about social acceptability and other such nuances. To record this information, Jane goes back to the list of samples and presses the callout button. This opens up a dialog box (Figure 16n) where Jane enters the information she wanted to record. By following this process for every sample, Jane has not only collected a few samples for her "play" gesture but also tagged each sample with relevant contextual information.

6.3.4 Two Mediums for Communicating Context

As seen above, Comoge supports two mediums for capturing and communicating context information: 1) Video with audio; 2) Text labels and comments.

6.3.4.1 Video of Gesture Performance

Comoge records a gesture in two complementary formats: sensor stream and video. While the sensor stream is an input to the recognition backend, video establishes the ground truth and is exposed to the user. Although the primary objectives of a video are to communicate performance of a gesture and capture any variations that occur naturally, the richness of the video along with audio also allows us to learn about the context in which it was recorded. For example, you could differentiate an example recorded on a street from one recorded in a restaurant simply based on ambient sounds. Comoge allows a designer to capture either a first person or a third person view (placing a camera on a tripod) of a gesture performance.

6.3.4.2 Text Labels & Notes

The text is how most of the contextual information is captured and how it is manifested in the UI of the tool, sometimes accompanied by iconic representations. Much of the text is generated automatically, providing labels of contextual information tagged onto gesture examples. The text is also a means of capturing any field notes a designer inputs to aid recall of social, and cultural aspects observed in-the-moment that cannot be sensed automatically.

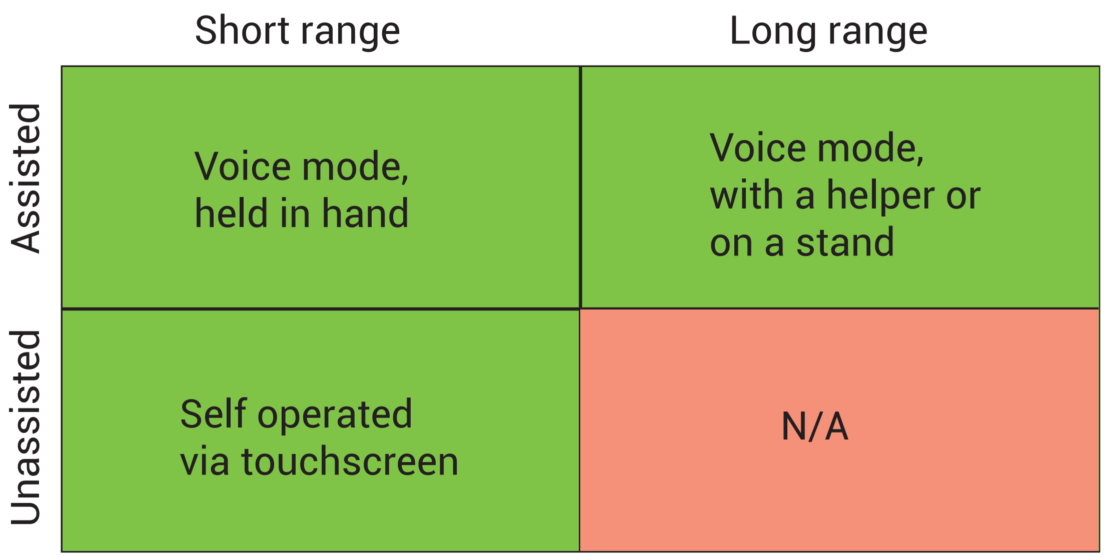


Figure 17: Chart showing different modes of recording sensor and video data. Green color indicates that such a mode would work well with Comoge. Red indicates that operating the controller in this mode would be difficult.

6.3.5 Modes of Recording Sensor Data with Video

There are two variables that define modes of operation with the controller for recording sensor data as well as video (Figure 17). These two variables are defined below.

Distance of the user from the controller: This variable can vary from short range (at an arm's length) distance from the controller to long range distance (more than an arm's length away from the body) to the controller.

Role of the person using the controller: If the main user himself/herself is using the controller while recording I term that mode of operation as unassisted. However, if the main user is operating the controller through voice commands or if a person other than the user is involved in operating the controller then I term that mode as assisted or voice command.

I have observed that Comoge works with unassisted or assisted mode in short range; however, the user would require extra support by another person or through voice commands when trying to operate the controller over long range distance. For close range, using assisted or unassisted mode depends upon which body part is being used for actual data recording. The idea is that extra movements should not be added to the sensor data pertaining to user interaction with the tool.

6.3.6 Adding Context to Motion Gesture Design

There are three approaches to integrating contextual information into the motion gesture design process: 1) bottom-up freeform exploration, 2) top-down planned execution, and 3) mixed.

6.3.6.1 Bottom Up

Bottom-up refers to a free exploratory approach to gesture design, where a designer first starts with a less constrained scenario then narrows down to identify what data needs to be collected and analyzed as they better understand the contextual aspects. For example, while designing smartwatch gestures for a music player application, she might want it to work in a wide variety of activity contexts and for different body placements. But during data collection, she realizes that testing against all these scenarios is not ideal as is evident from poor test results. Perhaps she needs to individually train and test in these various scenarios or increase number of samples with the complexity of a scenario. Another outcome could be ruling out certain situations as uncomfortable and improbable, thus focusing on a few important ones.

6.3.6.2 Top Down

Contrary to bottom up, the top-down approach refers to advance identification of important contextual aspects and putting in place a plan to collect examples for and test a gesture in all situations where it is intended to be used. For instance, taking the same example of smartwatch gestures for a music player application, the designer may want it to be operable while still, or on foot, and not while driving. Moreover, she may want it to work irrespective of the hand the watch is worn on. A tool could support this planning by providing a multi-step wizard that helps her think through possible variations of contextual aspects and lets her specify what is important to her scenario. With the collected information the tool can automatically prepare a conservative plan of where and how much data collection needs to happen. For example, it might suggest collecting 5 examples with each hand for "Wave *while still*" and 10 examples for "Wave *while on foot*" for a total of 2 (hands)x15 (still + on foot) gesture examples.

6.3.6.3 A Mixed Approach

While a purely bottom up approach is pure exploration driven, a top-down approach is a planned study. In reality, a designer seldom follows a linear path and, therefore, I see a mixed approach involving some bottom-up exploration followed by a more rigorous top-down planning phase. This is akin to what a pattern recognition expert does too. Alternatively, when a designer starts with a tentative plan and while collecting data to test her hypothesis finds outliers, she performs an analysis of the data collected to update her plan.

6.3.7 Using Context within Gesture Recognizers

Besides aiding recollection and reflection by designers, contextual cues also play an important role in the training and evaluation of gesture recognizers. For training, context helps automatically organize collected data into closely related bins so that separate recognizers can be built for each of the bins. For example, examples of the *wave* gesture collected while running can be separated from the samples collected while sitting. In this manner, contextual information can be used as constraints. A designer can then prioritize certain contexts over others to build narrowly focused recognizers that perform well in a specific context. She might choose to recruit remote participants for collecting more samples for a specific context using a tool similar to CrowdLearner [3]. Conversely, a designer might choose to relax constraints if she intends to build a general purpose recognizer.

6.3.8 Implementation

Comoge is written using the Android SDK v6.0. The main application, called *controller*, runs on a Nexus 6 smartphone. This application supports Bluetooth connection to any sensing device with a 3-axis accelerometer and a 3-axis gyroscope. I only tested an Android Wear *smartwatch* (Sony Smartwatch 3), a *head-mounted device* (Google Glass), an Android-based *smartphone* (Nexus 4), and a *custom* sensing platform (Metawear[1]). Each sensor

[1] `https://store.mbientlab.com/product/meta-motion-r`

is sampled at an average sampling rate of 50 Hz. Data is stored in the JSON format on a database on the phone.

I extract the following features per sensor channel (72 total): root mean square (RMS), standard deviation (SD), mean, normalized sensor energy, and ECDF [51]. As a proof of concept, Comoge uses an implementation of Support Vector Machines (SVM) available in the Weka library [50] for Android[2] as it has been shown to perform well for several classification tasks [17].

I initially tried the Activity Recognition API provided with the Android platform for inferring the common activity contexts such as biking, running, walking, and standing still, but switched to manual entry to overcome long response times (sometimes more than 10 seconds) and inaccurate prediction results (for example, holding a device up while walking is confused with being in a vehicle). With the goal of automatically determining location labels, I did preliminary testing with the Android Location API, but it requires a priori marking of geofences around places of interest which I found inconvenient and unreliable.

6.4 Evaluation

I conducted a summative evaluation of Comoge to determine its usability, and ability to support rapid prototyping of gestures, both indoors and outdoors. As described in more detail below, each participant used my tool to design and test gestures with wrist-worn (smartwatch) as well as head-mounted (Google Glass) wearable devices in three different activity contexts: a) still, b) on foot (walking), and c) in vehicle (riding a bus).

6.4.1 Participants

I recruited 17 participants (4 male, 13 female) and 2 pilot testers (S1-2), 5 professional interaction/experience designers (P1-5) from 3 design firms in Atlanta city, and 12 students (S3-14) from Masters in Human-Computer Interaction program at Georgia Institute of Technology. The participants had varying levels of expertise in Interaction Design, Programming, working with Sensors, and Machine Learning (Figure 18). I compensated each

[2] `https://github.com/rjmarsan/Weka-for-Android`

	novice				expert
	1	2	3	4	5
Programming	P10	P3, P6, P7	P13, I2, P5, P11, I3	P4, P9, I4, I5	P8, P12, P14, I1
Working with Sensors	P6, P13	P4	P3, P5, P11,	P7, P9, P10, P12, I3	P8, P14, I1, I2, I4, I5
Machine Learning	P3, P4, P9, P11, P13	P5, P7, P10, I4	P8, P12, I3	P6, P14, I1, I2, I5	

	<1 year	1-3 years	3-5 years	>5 years
Professional Interaction Design	P3, P5, P10, P11, P12, P14	P4, P7, P9, P13, I3, I4	P8, I1	P6, I2, I5

Figure 18: Summary of prior experience of the study participants along four dimensions: programing, working with sensors, machine learning, and professional interaction design.

participant with a $20 gift card.

6.4.2 Procedure

I had a three-part user study which took 2 hours (30 min. each for both part 1 & 3, 60 min. for part 2) to complete on average. In the first part, I started out by defining motion gestures, and the higher level purpose of my tool for my participants. After getting their informed consent, I provided them a five-minute demonstration with a representative task of creating two smartwatch gestures for an activity tracker application. Participants were encouraged to clarify doubts during the demo or while they were exploring the tool themselves. After freeform exploration for a few minutes, I assigned them their first task to be completed indoors while still. I provided them a printed sheet which asked them to create a project “Combat With Fruits,” design three smartwatch gestures for “Shoot(ing) a melon," "Slice(ing) a pineapple in air," and "Smash(ing) a coconut on table" respectively. Then they were asked to train each gesture thrice at least and conduct at least three tests. During exploration and first task, I followed a think-aloud protocol, took notes, and recorded video.

In the second part, participants were provided another printed sheet with a design brief for a task to be completed outdoors in two different activity contexts: while walking, and while in a bus (when not possible, replaced with other outdoor scenarios to introduce variety). They were asked to first identify four common operations for a music player running on a smartphone, then design gestures for each operation for both a smartwatch and Google Glass, train the gestures at least thrice in each of the two activity contexts mentioned before, and test the gestures as many times as wanted. In total, this task required participants to design eight new gestures, two (one with a smartwatch, one with Google Glass) per operation. For the walking scenario, I took participants to different public places near my campus or their workplace. For the bus scenario, I boarded my campus public transport (or an escalator for a few participants). I sat down in an accessible location during the walking scenario whereas in the bus scenario I accompanied them often sitting/standing next to them. In both cases, I took notes and pictures.

For the third part, I went indoors with the participant, where I first asked them to fill out a multi-part questionnaire for measuring usability of the tool and collecting demographic information. Then I reviewed the gestures that the participant designed with them, followed by a semi-structured interview. I video recorded the review and interview sessions and took notes. At the end, participants were compensated and thanked.

6.4.3 Data Collection & Analysis

I collected several streams of data throughout the study. As mentioned above, I recorded video during indoor sessions,and took pictures while outdoors. The researcher conducted observations and took notes during the entire time. On the device running my application, I collected interaction logs, app data including a backup of the sensor data, and videos that the participant recorded with my application. Additionally, I recorded responses to the questionnaire I administered. My questionnaire was based on the System Usability Scale (SUS) [15] and the Questionnaire for User Interface Satisfaction (QUIS) [26].

For analysis, all interviews were transcribed and analyzed with the corresponding observation notes to identify broad themes related to prototyping gestures in context which may/may not have been enforced or influenced by Comoge. Two researchers, I and a colleague, systematically analyzed this qualitative data to draw out novel issues/insights and also trends/patterns across all participants. We reached consensus through discussion of these findings. I extracted quantitative information about the quantity and description of projects, gestures, and gesture samples created by each participant from the backup of application data and compared that with the broader themes identified from the qualitative data to identify patterns and relationships. This was compared to the calculated duration of application usage for each task from the logs. I conducted statistical analysis of the SUS and QUIS.

6.4.4 Apparatus

I installed the Comoge application on Android-based Motorola Nexus 6 devices and companion apps for sensor recording on Android-based Sony Smartwatch 3 and Google Glass, all running latest version of Android. I used another smartphone mounted on a tripod for video recording of sessions. Participants filled the questionnaire on a web browser running on an Apple Macbook Pro.

6.5 Results

My main hypothesis with this study is that a standalone mobile tool can serve as a rapid prototyping tool for gesture design in ad hoc as well as naturalistic settings. Here I present results from the user study. I start by summarizing quantitative information from application logs and questionnaire responses. Next, I provide evidence for usefulness of designing *in situ*. Lastly, I share support for context-based prototyping by noting the impact of contextual factors on the use of Comoge and their gesture designs.

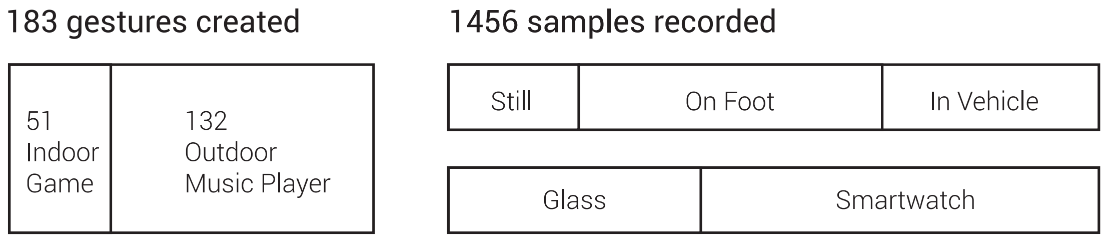


Figure 19: Summary of output from participants along with distributions of the samples recorded by activity context and form factor and distribution of gesture by location and application genre.

6.5.1 System Use & Designed Gestures

Figure 19 provides a quick summary of the output of my study participants. During the study, 17 participants designed at least 11 gestures each in an average time of approximately 59 minutes of system use (≈**4 min/gesture** indoors & ≈**6 min/gesture** outdoors) including planning time, time to walk out of building, waiting for bus, crossing streets, and unexpected interruptions such as greeting friends. On an average, every participant spent 11 minutes indoors where they recorded 16.4 samples total for 3 gestures, whereas they spent 47 minutes outdoors to record 69.2 samples total for 8 gestures. Although two participants didn't get a chance to try my tool in vehicle due to unavailability of bus service and escalators, on an average each participant recorded 17.2 samples *while still*, 39.94 samples *while on foot*, and 28.52 samples *while in vehicle*. As for the usage per sensor recording device, each participant on average recorded 52.4 samples for a smartwatch including indoor and outdoor use, and 33.23 samples for the Google Glass which was only used outdoors. One professional participant skipped Google Glass due to a health condition. The study results support the hypothesis that the tool allows rapid prototyping of motion gestures in naturalistic contexts while being context aware. In S7's words, "*It cuts the time in half in trying to code something and test it.*"

6.5.2 Tool Usability

As seen in Figure 20, on the usability questionnaire (with 0-9 scale for questions from QUIS, 1-5 for the rest) that I administered to each participant upon completion of both

	Score (1 - 5)			
	Students (14)		Professionals (5)	
	Avg.	Std. Dev.	Avg.	Std. Dev.
System is Easy to Use?	4.17	0.83	4.2	0.45
System functions are integrated?	4.25	0.62	4.2	0.45
System is Easy to Learn?	4	0.95	4.6	0.55
I am confident in using the system?	4.17	0.72	4.2	0.45
	Score (0 - 9)			
Organization of the information on screen is clear?	6.5	1.73	8	0.71
Sequence of screens is clear?	7.17	1.34	8.2	0.45
Exploring new features by trial and error is easy?	7.25	1.06	8	0.82
System is fast?	7.25	1.22	8	0.71
Error messages are helpful?	5.58	3.03	5.75	4.03
System is reliable?	4.92	2.31	6	2.35

Figure 20: Summary of questionnaire responses about the tool's usability.

tasks, my participants rated Comoge including companion apps high on ease of use (*scale*: 1-5, μ=4.17, σ=0.73), ease of learning (*scale*: 1-5, μ=4.17, σ=0.88), ease of exploring new functions through trial and error (*scale*: 0-9, μ=7.43, σ=1.03), speed (*scale*: 0-9, μ=7.47, σ=1.12), visual clarity (*scale*: 0-9, μ=,7.88 σ=1.21), and integration of various system functions (*scale*: 1-5, μ=4.24, σ=0.56), while rating it low on inconsistency (*scale*: 1-5, μ=1.7, σ=0.78). Unfortunately, few of my participants found the system error messages unhelpful (*scale*: 0-9, μ=5.63, σ=3.16) thus raising concerns about system reliability (*scale*: 0-9, μ=5.24, σ=2.31) in their minds. I believe Bluetooth connectivity issues to be the biggest contributor of unexpected errors. However, despite their limited exposure to my tool they had high confidence (*scale*: 1-5, μ=4.18, σ=0.64) in being able to use the system themselves, and clear understanding of the workflow (*scale*: 0-9, μ=7.47, σ=1.23). For instance - "*It was a little bit of a learning curve because it was a lot of steps at a time, but once I got the hang of it, it was really easy to just create new content*"(S5). Going forward my goal is to improve my system's performance on all of the metrics listed above, thus augmenting its overall usability.

6.5.3 Providing a Flexible Framework for Gesture Design In Situ

Typically an interaction designer does not have a streamlined process for gesture design and more importantly, is näive with respect to challenges pertaining to gesture recognizers. In the absence of a prototyping tool that encapsulates a workflow based on expert knowledge and features to lower the barrier, interaction designers resort to using of a custom pipeline comprising of multiple tools, thus spending precious time on tasks not central to the design exercise. Unfortunately, such custom solutions like prior tools do not fully support exploration in naturalistic settings and definitely not while also recording contextual information.

As far as I know, Comoge is the first fully mobile motion gesture design tool that works with commodity wearable sensors and leverages context to inform better designs and also provides a streamlined process, for novice designers to follow. This is further backed by remarks made by my participants during the interview. For example, "*...given someone who has not designed gestures that much. I was mostly learning as I was going.*" (S12).

6.5.3.1 Rapid, Iterative, & In Situ Design

The process provided by Comoge allows for rapid prototyping as supported by the evidence from the study where participants were able to generate a significant amount of content within a short period of time. As Comoge supports and encourages recording of gestures in situ to capture context. Rapid prototyping also inherently supports experimentation to see what works and then provide refinement. The benefit of such a design process is further echoed by participants, for example, S7 mentioned, "*It was really helpful to be able to test as I went along, that helped to quickly on the go eliminate some gestures.*" Another example is when S9 said, "*I think it is having the ability to record the gesture multiple times is good, especially when you know there are certain contexts that are pretty noisy.*" Similar remarks were made by P1 and S12.

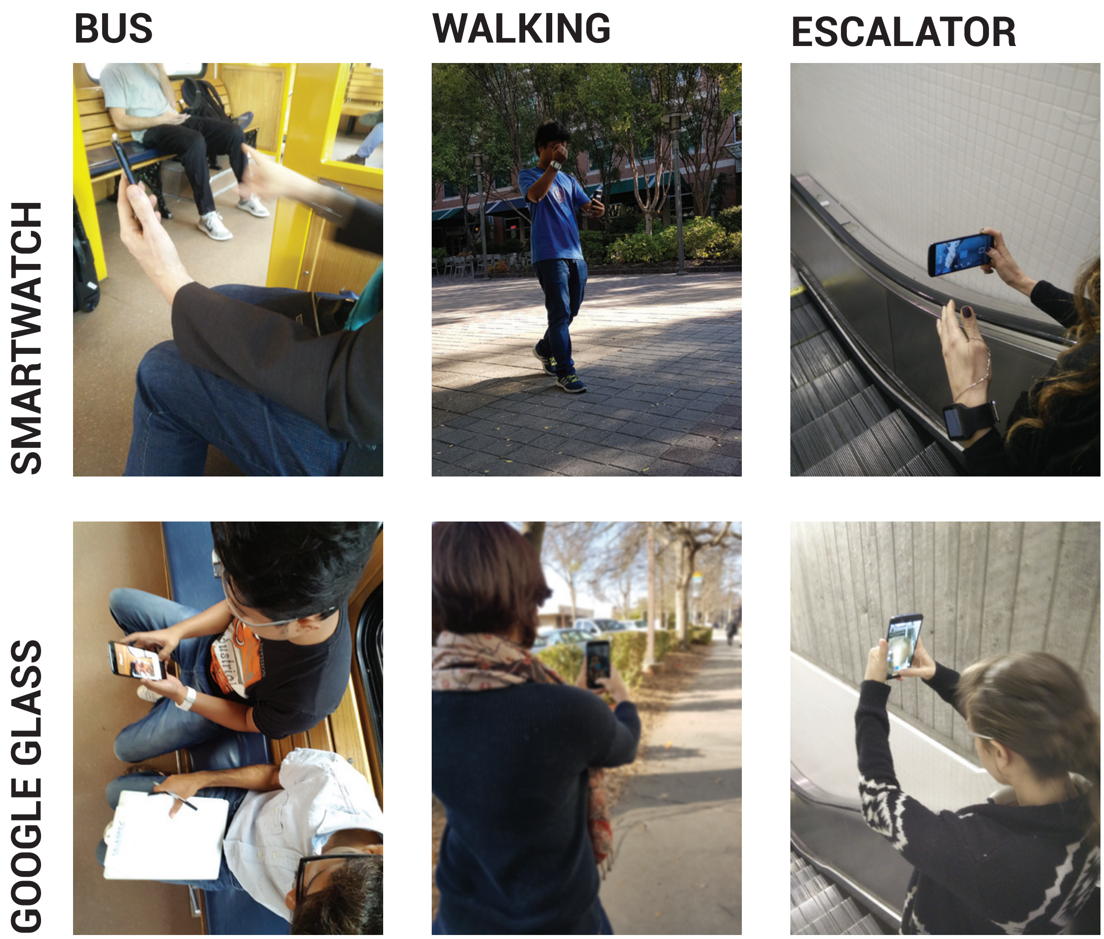


Figure 21: Comoge in use by user study participants. This 2 by 3 grid shows a sample of mobility and form factor settings that participants were exposed to.

6.5.4 Understanding Differences between Contextual Settings

As seen in Figure 21, the participants experienced a variety of contexts during the user study. From the data that I analyzed, I found that there were multiple instances in which designing gestures with Comoge prompted the participants to either comment/question their understanding of different contexts that they were in. The remarks made by the participants and my observations can be categorized as follows.

6.5.4.1 Activity Contexts

Comoge prompts users to record gestures for different activity contexts by categorizing the recorded gestures into respective activity context tabs specified by the users themselves. Visibility of other activity contexts for which there are little to no recorded gestures prompts users to record gestures for those activity contexts as well. "*I liked it gave me the 3 scenarios beforehand - while in vehicle, still and while moving. I wouldn't have known that the 3 are different if I wasn't told this beforehand*" (S13). In the study during the outdoor task in which the participants had to design gestures in a moving bus, I observed that most participants waited for the bus to move before recording gestures to make sure they capture the motion of the bus also as a part of the gesture. One of the participants even pointed it out saying, "*I think I should do it while the bus is moving*" (P5) When some of their gestures failed to be accurately recognized while testing in the bus they were able to comment saying, "*...I think that's because the vehicle is actually, it has some more, like vibration, and also it has movement, it has acceleration, it has turning right or turning left...*" (S12). Similar remarks were also made by several other participants (P5, S6, S8, S9, S13).

Participants were also curious how walking or being in a moving vehicle would affect the gesture recognition. "*…Things like horizontal motion, what are the factors that differentiate a still mode to an in-vehicle mode...*" (S13).

Lastly, participants mentioned that designers would want consistency in the gestures they design, that is, they should work no matter which activity context the user is present in. "*I kind of designed gestures so that I don't have to change it irrespective of where am I. I don't think it is good for user oh I am in a bus I should do this, oh I am walking I should* *do this.*" (S8). A similar remark was made by S10.

6.5.4.2 Devices & Sensors

From my analysis, I found that the participants were also concerned about how the sensors worked and were prompted to think about different perspectives based on the device and sensors they were using. For example, questions like, "*Will it measure impact, is there a difference between (thumping) or moving in air only?*" (S7) or "*You have to understand the limitations of the sensors, the granularity of the sensors data*" (P5) clearly suggest that the working of the sensors was something that some of my participants were considering when designing gestures. Other than the participants already mentioned, (P3, P4, S8, S11) also raised similar questions.

Furthermore, the tool was also helpful in communicating the limitations of devices. When recording samples, my participants quickly recognized that a particular device offers only a few affordances. This is important as it provides the designers with constraints which they cannot violate with their gestures. For example, the participants commented saying that, "*it is a very good tool for you to be even be able to project what it is you are trying to make, how far it can go, because even with testing it for 10 minutes on Google Glass I was like, in my head you can do this, you can do that, oh wait I am just like limited by 4-5 things. it really helps you to scope out.*" (S7). Similar remarks were mentioned by P4, S8, and S14.

6.5.4.3 Body Locations

Another category that emerged from my analysis, was concerns/prompts related to body locations involved in performing the gesture. My participants mentioned things like "*I feel like for Google Glass because your head can move only so many ways the gestures are pretty limited whereas apple watch there are a lot more things that I can do...*" (P4). Similar comments were also made by S5, S9, S13, and S14. This comment shows that the participants were actively thinking about the differences between body locations and their limitations. Furthermore, the participants realized that different body locations require different gesture samples to make sure that gestures work even if the device is worn in another similar position but on a different limb (for example, wearing the watch on the left hand vs. the right hand).

The above examples clearly show that at least activity, device, and body location contextual factors were important criteria for the design of gestures for my participants and they were able to take those aspects into consideration due to or with the help of Comoge.

6.5.5 Changing Gestures based on Contextual Constraints

During the interview, I also reviewed the gestures that were designed by the participants to capture important information about what the participants were thinking while they were designing a gesture. A major insight that I gained from the analysis of that data is that the participants were compelled to change the design of the gesture to something else based on the constraints enforced by their surroundings or by the devices. The role of Comoge in facilitating recording of samples by designers for different contexts with different devices at different body locations is crucial for effecting such changes in the design of gestures. There were several participants who considered changing their gesture based on either activity, device/body location or social constraints. For example, S10 said, "*I want to like draw a large like a wave in air but then thought that it may not be like socially acceptable... so I changed it, like, you could shake your hands in your pocket.*" Another example is when S9 said about a tapping gesture, "*So initially I began with a single tap, but tap is so easy to get incorrect because you are tapping a lot of places even though you don't have the intention of playing or pausing any song, that was the reason I made it a double tap.*" Another example is when P1 said "*...that was my embarrassment, causing me to change my behavior. It was interesting, I never thought of that before.*" Similar comments were made by P3, P4, S8, S9.

These examples show that knowing about the context and the constraints that arise due to them makes a significant impact on the design of the gesture. Such changes could be easily missed if the same gestures were being designed without the context in mind.

6.6 Reflection on Usability, Appropriateness, & Recognizability of Gestures

I asked participants three interview questions, among others, seeking their reflection on the usability, appropriateness, and recognizability of the gestures they had designed in past one hour. The variety of responses I received indicate the usefulness of the tool and study design in bringing these three important metrics to the foreground. In this way, the tool acted as a medium of reflection. Below I present tabular summaries of participant responses along with some quotes for each criterion.

6.6.1 Usability

In the absence of a standard definition for the usability of motions gestures, I started with the notion of *ease of performance*. Hence, I posed a question to the participants about ease of performance of their own gestures. The following table summarizes their responses.

Do you think the gestures you designed are easy to perform?

Yes(all)	*No(all)*	*User testing needed*	*Some*
11	1	3	2

Table 6: Summary of participant responses to the gesture usability question.

As can be seen in Table 6, only 3 participants (1 professional & 2 students) chose to defer a judgment until they user test their gestures. Ideally, I would expect all participants to respond this way because if a gesture is easy to perform for them does not mean others will feel the same way. However, since they had performed each gesture several times in multiple mobility conditions, their assessment of usability is arguably more credible than with a one-shot experience. Consider, for example, the case of S4 who first answered yes and then pointed a caveat: *"Would I want to do it 20 times a day? Probably not."*

Participants not just reflected on usability they helped us define usability. As pointed in the quote above, the frequency of use of a gesture is an important factor when evaluating its usability. Similarly, participants also identified cognitive effort, memorability, and seriousness of the action connected to a gesture as other contributing factors. Here is what S13 said, *"I'm associating 'easy' with requires less physical and cognitive effort. And what*

I mean is doing the gesture again and again would not tire someone.". S5, who answered *some*, also remarked that *"it is the repeated use over time and especially for longer gestures remembering what they are."*

Finally, I will like to note that most participants found head gestures harder to design and perform when compared to hand gestures. I offer two reasons for this: 1) they were unfamiliar with head gestures and 2) head provides lesser degrees of freedom and, if moved fast, can cause unpleasant side affects such as dizziness.

6.6.2 Appropriateness

Appropriateness of a gesture is predominantly defined by its social acceptability. However, other contextual factors including mobility, location, cultural setting, and the form factor as well as the body placement of the device also contribute to appropriateness. Therefore, I asked the participants to comment about the social acceptability of their gestures and also to compare and contrast their experiences designing gestures indoors and outdoors. The following table summarizes their responses and subsequently presented categories collate their qualitative remarks.

Do you think the gestures you designed are socially acceptable?

Yes							No	Not All
In US	*In Tech*	*For Head*	*For Hand*	*For Me*	*While Walking*			
1	1	1	3	1	1	6	0	3

Table 7: Summary of participant responses to the gesture appropriateness question.

Table 7 suggests that all participants had considered social acceptability of their gesture designs, but only half of them could delineate their concerns. For example, one professional participant noted that her gestures would only be acceptable in the USA. Others qualified their primary response of *yes* with "In [Georgia] Tech", "For Head" or "For Hand" gestures, "For Me", and "While Walking". Remaining 6 participants didn't qualify their "Yes" response. But the 3 participants with "Not All" responses provided examples of gestures they may consider unacceptable for reasons shared below.

6.6.2.1 Social Acceptability

Based on the experience of the participants they provided us comments about situations in which they thought that the gestures they designed might be weird or odd. S9 said, "*I was conscious of the fact that I shouldn't be directly anyone into the eye when I am doing this gesture or that gesture (head tilting gestures).*" P3 said, "*I was thinking of it as I was doing it. I was thinking "Oh God..." cause I have seen people do that and it freaks me out.*" Similar comments were made by S6, S7, S8, S10, S11, S13, and S14. Participants also commented saying that the location or the societal setting in which the gestures are performed also influences what was socially acceptable, "*... like at Georgia Tech, gestures are socially acceptable, yeah. At a country diner, it may be a little weird*" (S11).

I also identified that my participants weren't worried about what others were thinking while they were still or walking but they were aware of their surroundings in closed spaces like the bus. For example, P4 said, "*...if you are walking down the street and you just start going like this (performs the gesture) what does that mean to someone?*" S5 also said the following, "*...social acceptability wasn't something much of a factor for me, till being on the bus happened.*" S13 also echoed similar sentiments.

It is important to note that the participants did not make use of Comoge's feature to take notes in-situ to record these observations that they had made while in that context. This suggests that the ability to capture social context needs to be re-thought or maybe the tool needs to draw more attention to it. Perhaps, Comoge should support recording of voice memos instead.

6.6.2.2 Managing & Manipulating the Perception

The participants also commented on ways in which devices used for performing gestures help in managing and manipulating perceptions. For example, S6 said, "*Felt locked into my phone allows people to judge that you are in your own world. Would have felt weird moving head without the phone.*" A similar sentiment was also echoed by S7 and S9 who mentioned that wearing devices acted as a shield for them. For example, S7 said, "*I felt more confident in doing what I was doing because I had a Google Glass on...*".

Some participants also mentioned that with time gestures will become more and more acceptable. For example P5 mentioned, "*...this cool new thing came out but you have to like do this, you see more people doing it and the more people that do that the more normal it becomes...*" Once commodity products start supporting gestures inherently, then the adoption of gestures among society would be faster and higher. S5 and S7 raised this point based on history with acceptance of people talking on the phones or texting while walking.

6.6.2.3 Looking for an External Perspective on Designed Gestures

When asked about the social acceptability of the gestures that the participants had designed some participants mentioned that they would need critique and feedback from external observers to be able to comment on that. For example, S4 said, "*I think it deserves some objective perspective like critique from an external perspective.*" P2, S5, and S7 also made a similar comment. S8 also mentioned "*I try to think about things which people do and I don't mind seeing people*" when talking about designing gestures that are socially acceptable. This notion of social acceptability that is reliant on both first-person (user) and third-person (spectator) perspectives is in line with Montero et al.'s definition [97].

6.6.3 Recognizability

Recognizability is conventionally defined in terms of a gesture recognizer's ability to recognize a learned gesture from an incoming stream of sensor data. Comoge's machine learning backend performed well on this metric. As S5 said, "It's ability to recognize was better than I had thought it to be." Most other participants were similarly impressed by Comoge's ability to recognize gestures based on a small sample of training data. Obviously, in some cases, Comoge didn't correctly predict a gesture's label. In these cases, the participants performed real-time analysis and devised strategies to avoid misclassifications. Here is one from S8: *"Coz when you walk on the street or when you are on the bus it is very usual if you just look around so I can't use these actions to use them as an input to control so I first, all of them begin with a nod."* Additionally, participants used repetitive motions to disambiguate a gesture. Unlike in Ashbrook's [7] case, my participants did not use *impact* as a strategy.

Because the second task was outdoors, the participants also assimilated a social meaning into the definition of recognizability. Besides machine recognition, they were also concerned whether a person approaching you or just observing you at a distance would *recognize* that your gesture is not directed towards him/her. This led participants to move away from gestures with meaningful associations in human-human communication such as nodding and also consider subtle gestures. For instance, P2 said "Stop could be like no, to someone, yeah like, don't come, if someone is coming towards me, yeah." P3, P4, S5, and S12 made similar remarks.

Several researchers in the past have focused on the creation of more accurate gesture recognizers but it isn't the primary focus of my research. Therefore, rather than asking a direct question about recognizability of their gestures, I instead asked participants to comment whether their gestures will be confused with some natural movements by a recognizer and, more importantly, by a person? Table 8 summarizes their responses. Evidently, 7 participants deemed their head gestures not to be uniquely recognizable. This relates to the observation I made during the discussion of the usability criteria about their challenge in imagining new head gestures.

Do you think any of the gestures you designed will be confused with some other natural movements occurring throughout the day?

Yes(one or more)	*For head gestures (e.g., Stop, Nod)*
10	7
By recognizer?	*By other people?*
13 (S5)	5 (S5)

Table 8: Summary of participant responses to the gesture recognizability question.

6.6.3.1 Postulating Possible False Trigger Scenarios

Furthermore, I found that inviting the participants to an outdoor context and moving contexts during the study made them think about false trigger situations that would arise as a result of on the go movements. For example - "*...suppose I am in a bus and the bus driver applies brakes and I have to raise my hand to hold something and my system identifies this as a start*" (S8). P4 and S14 also raised a similar concern about designing gestures for walking context and realized that gestures designed for that activity should not be confused with swinging movements that generally are part of the walking motion.

6.6.3.2 Interactive Testing

Related to the original notion of recognizability is the result of testing a gesture. In Figure 22, I offer a plot of number of tests performed by each participant with a clear demarcation of tests that resulted in an incorrect label in red color. Interestingly, the reader might recall from the study procedure that the participants were asked to perform at least three tests for the first task and any number of tests for the second task. However, Figure 22 clearly shows that most participants tested many times over, sometimes going as high as 44 times (for example, S8). Therefore, the results are a coarse indication of the recognizer's performance on the novel gestures that the participants designed during the study. But I caution against reading too much into these numbers as the evaluation of the recognizers trained by the participants, with few examples, should be performed using standard machine learning methods such as cross validation. I leave that for the future work where a larger set of examples for a smaller set of gestures might be collected.

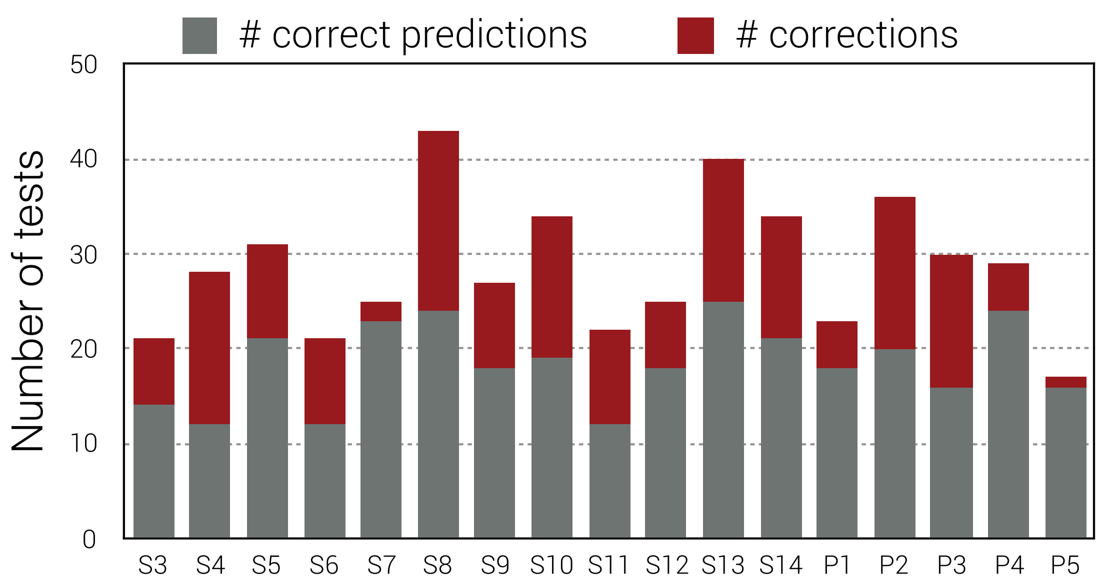


Figure 22: A plot of number of tests performed by each participant with a red highlight of the proportion of incorrect predictions produced by the recognizer which were ultimately corrected by the participant.

Before moving to how the corrections of labels were facilitated, I want to list down the four factors that contributed to the prediction errors. First, *high variance within examples* of the same gesture lead to poor recognizability. For example, a participant tried slicing a pineapple in all possible orientations such as vertically, horizontally, and even diagonally. Second, *similarity between gestures* for a project caused confusion and low goodness scores. For example, slicing a pineapple vertically and smashing a coconut looked similar if the only difference was an open palm in one case and a claw in another. Third, Bluetooth connection losses while recording led to crashes which in turn rendered recognizers untestable. I had to ultimately rebuild the models to resolve the issue but sometimes it was too late until I noticed. Finally, because I had designed the application to handle computationally heavy tasks such as goodness score calculations in the background, the multithreading approach failed me when the user jumped too quickly between screens before the calculations could complete. Unfortunately, participants experienced few crashes due to this reason.

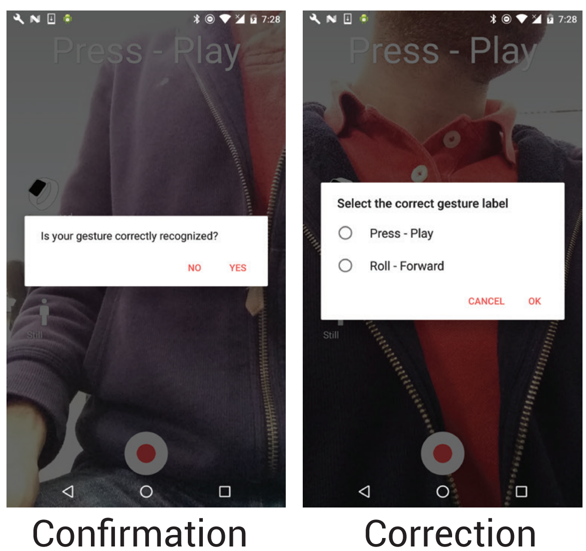


Figure 23: Two screens from Comoge that demonstrate human-in-the-loop approach where the users can confirm or correct recognition results.

From the beginning, I was aware of the limitations of the best-designed machine learning systems, so I intentionally chose a human-in-the-loop approach. I wanted to create a dialogue between the participant and the recognizer, in a manner similar to *Crayons* [39]. As shown in Figure 23, after every test the application confirmed the test label with the participant and also allowed him/her to correct the label. Based on a participant's response, the labeled example then becomes part of the training data for the respective gesture and the models are rebuild. Not only does this interactive approach allows the application to utilize every example a participant provides, it also gives the power back to the user.

6.7 Discussion

So far I have elaborated on the findings from the user study which I was anticipated, but now I will share some unexpected insights I gathered. I will start by noting the differences between the student and professional participants. Next, I will share a concern many participants had regarding the basic premise in my study design; designing outdoors is necessary.

6.7.1 Differences between the Professional & Student Designers

Although most of the participants were novices with respect to motion gesture designs, I noticed a few differences between the professional and student designers. First, professionals were more critical of the UI of the tool. Second, professionals were concerned with integrating the tool into their existing workflow.

6.7.1.1 Focusing on the UI of the Tool

All of my professional users were iPhone users who used the tool on Android. Furthermore, being professional designers, in their daily work focus more on designing and building digital interfaces for the web. As a result, I observed that the professional designers paid more attention to the UI design and issues with the Comoge tool whereas the students paid more attention to gesture design and issues arising from that process.

6.7.1.2 Streamlined Workflow

Another aspect of the difference between the professional participants and the student participants was that the professional participants were thinking more about integrating the tool into their everyday design workflow. I received comments about using the tool for user research, gesture design and even for initial development. For example, P1 said, *"I think also I would love to strap this (watch) on a user and ask them how would they* *design. Give them the functions that I want them to perform and see how they creatively interpret the label... If I had a sample set of like 15 people and 80% of them use the same gesture, then I can go ahead and design for that gesture confidently.*". The same participant also mentioned that Comoge could be used as tool to provide the development team with a deliverable from which they can start their work and reduce miscommunication at the same time - "*If I as the XD designer could perform the motions that I wanted to perform and give that data packet to my developer and say instead of you having to interpret my drawings, here is that. just plug and play.*" Similar remarks were made by P3 and P5.

6.7.2 Why Outside?

Although there is strong evidence in the favor of in situ and context-based prototyping, surprisingly, the participants showed a lot of hesitation in recording gesture examples outdoors. P5 offered her view that "*When you are designing you have to capture lot of stuff make mistakes and try it again, it is very very messy.*" Her point is that a designer is more comfortable sharing the end product of the design process. Suggesting a different challenge, S4 remarked that "*It took [me a] while to get into the mindset of ignoring people gawking at you and to focus on designing a task.*" S7 and S6 made similar comments. My knee-jerk response is that their reactions are a side effect of the study design where they were forced to start the design process outdoors. In real life, they will have a choice to do as much indoors before testing their gesture designs outdoor, which they should. A comment from P1 provides support for this hypothesis. She said "*I don't know if the ability to record (video) for the first time in that environment does us any more good as XD professionals than train it in an indoor closed environment.*"

However, reflecting on my experience in the pilot study suggests a deeper issue here. In my pilot tests with two participants, I asked the participants to recruit one random person and show them how to use the tool and then ask them to perform a gesture designed by the participant. Even though we did this within my academic building where people knew each other, the participants strongly questioned the motivation for this task. One could argue that perhaps this is an opportunity for designing a script for the participants that would make them comfortable in recruiting participants for a quick sanity check of their gesture designs. Alternatively, I argue that there is a need to educate the interaction designers working on gestural interfaces about the unique challenges that it entails because a motion gesture is seen as a public performance.

6.7.2.1 Perception of a Shield

To the point about gestures as public performance, some participants reflected that the uncommon devices they had to wear, the visible (and distinctly audible) nature of the application helped them save face when in public areas. For example, S5 said "*we were in a space where people kind of understand that we are doing testing or they saw that I had a device on, and the app was chirping so they knew something was going on.*"

6.8 Limitations & Future Directions

Major issues that I identified with the tool design based on participant responses can be broadly categorized as follows:

6.8.1 Recording a Video while Performing a Gesture

Recording a video of a gesture performance though a useful feature was also one of the biggest usability concerns for most of my participants. In my gesture recording and testing screens, I currently require the participant to record the video to capture the context in which the user is recording the gesture sample. However, the cognitive load involved on part of the user is high as they are required to start the recording, perform the gesture, end the recording all the while making sure that the video is capturing enough data. A participant reflected, "*Recording was an issue with the watch because it was difficult to coordinate things*" (S8). S7 also remarked that "*you tend to get distracted by the fact that there is a camera there because you are thinking the camera is also sensing it, but actually, the camera is not sensing it.*"

I anticipated these issues, so I enabled other modes of operation of the recording module which are characterized by two variables: (1) distance of the user from the controller; and (2) role of the person using the controller. The distance could be within arm's length (short) or beyond it (long). The role could be independent or assisted. My participants only experienced the tool in short range via touchscreen and independently. But Comoge also supports speech commands with which the tool can be operated in short range or at a distance (for example, by mounting the controller on a tripod or asking a colleague to hold it for you). Alternatively, I could make video recording optional during a gesture performance. I may instead provide an option during gesture creation to record a video as part of the description.

6.8.2 Understanding of Machine Learning

From the study, I observed that participants who had a better understanding of machine learning were able to learn the tool more quickly and also understand what is happening in the background without much feedback. However, participants who were novices in machine learning struggled a little with the tool. They had questions regarding how their motions and gestures would affect the recognizer. my participants asked us questions like "*What is a good number of sample size? How many distinct samples do I need for it to say oh you're good?*" (S13).

Comoge is not designed as a walk-up-and-use system. Similar to other tools, it requires some tutoring for gaining proficiency. I are looking forward to exploring novel guidance and feedback mechanisms. Long et al.'s [89] work in providing automated advice for users of a stylus-based gesture recognition system is an illuminating exemplar.

6.8.3 Lack of Guidance

Related to the point before is the issue of guidance. Comoge wasn't designed to give explicit instructions or provide recommendations to participants. Instead, it is designed for open-ended use by the designers interested in testing a variety of gesture ideas in multiple contextual settings. Having said that, I agree that at times an inexperienced user can get lost without a clear direction to proceed on. For example, S13 asked "*What is a good number of sample size? How many distinct samples do I need for it to say oh you're good?*" But the application didn't offer a response. In retrospection, I will like to point out that Comoge provided implicit guidance in the form of goodness scores, tabs for each activity context, and feedback on the recording screen but could have done better on explicit guidance. It could suggest, for example, the number of samples needed for a particular activity context. Moreover, it could expose inter- and intra-class variances similar to the *MAGIC* [6]. Or provide recommendations like *Quill* [89]. However, in the wake of evidence from both of these systems that suggests the extra guidance and recommendation wasn't very effective, this topic of guidance requires a deeper investigation in future.

6.8.4 Reliability of the Tool

During the study, there were some problems in terms of Bluetooth connectivity between devices, especially Google Glass, which resulted in system crashes which further affected the training of recognizer models. As a result, the gestures designed by the participants weren't always accurately tested. my participants performed 465 tests in total out of which 166 resulted in a wrong prediction, which they corrected immediately. I attribute only a portion of these errors to the Bluetooth issue, the rest were genuine user and recognition errors. In future versions, I will make the recognition system more accurate and the tool more robust.

6.8.5 Other Considerations

6.8.5.1 Automated Body Position Sensing

Comoge currently doesn't support automatic sensing of body parts used to perform a gesture. Although the tool visually reminds the designer of the intended body part on the gesture training and testing screens, it would be better to automatically sense and prompt the designer to correct the placement if necessary. One possible solution is to ask the designer to perform examples of a gesture that is most representative of the intended body part as a warm up exercise. For example, to distinguish which wrist the designer is wearing a smartwatch on, the tool could suggest checking the time three times. Alternatively, the tool could just ask the designer to confirm the body location. An added advantage of automated sensing is that the tool could automatically tag each sample with the sensed body placement, thus supporting a bottom-up capture approach.

6.8.5.2 Connecting Gestures to Real World Applications

Currently, Comoge is equipped to tell the designer if their designed gesture works or not. However, it would be better for the purposes of prototyping if I could connect the gesture to a template/real example application to make sure that the gesture works in the context of its use with the application and that the experience is as assumed. This can be accomplished by connecting gestures with services such as IFTTT[3].

6.8.5.3 Automatic Gesture Recognition

Currently, to test a gesture in Comoge the user has to provide a trigger in the form of a button press to tell the system to "start recording" and "stop recording". However, a real-world use case of any gesture would not involve such triggers. Therefore, supporting automatic recognition of gestures without such manual delimiters would be a valuable addition to the feature set of Comoge.

6.9 Summary

In every interactive system, *context matters.* The capture, access, and use of context are critical to the design and prototyping of usable and socially appropriate motion gestures. Here, I present a standalone mobile motion gesture prototyping tool called Comoge that leverages six types of contextual cues, including activity, location and time, sensing device and sensor, body placement, social and cultural factors, and application state. Moreover, I share the results from a user study which supports Comoge as an easy to use and learn system that empowers designers, with different levels of expertise, to prototype motion gestures rapidly in context. By visually presenting contextual information my tool also enables better reflection on and critique of gesture designs.

[3] `https://ifttt.com/`

CHAPTER 7

CONCLUSIONS & FUTURE WORK

This thesis aims to advance our understanding of why and how to support the rapid design of appropriate, usable, and recognizable motion gestures in naturalistic contexts. To this end, I first outlined a design space characterizing the factors impacting the design of motion gestures, a designer's process, and the design of potential tools they would use (Chapters 3 & 4). Subsequently, I presented my explorations of how to enable *in situ* and context-dependent design of motion gestures through two iterations of a smartphone-based tool (Chapters 5 & 6). In this chapter, I first restate my thesis statement, research questions, and contributions, then begin to generalize from my experiences by proposing guidelines for the design of future tools and by identifying open challenges and opportunities for future research. Finally, I conclude with a summary of this research.

7.1 Restatement of Thesis Statement, Research Questions, & Contributions

My thesis statement is as follows:

> **A mobile motion gesture design tool that supports in situ, context-based prototyping will enable rapid creation of gestures for wearable/handheld devices and reflection on their usability, appropriateness, and recognizability.**

In this thesis, I considered following three research questions.

RQ1. *What design guidelines and process do designers currently follow while working on a novel motion-based gestural interface?* (Chapter 3 & 4)

RQ2. *Can a mobile motion gesture design tool support rapid in situ prototyping of motion gestures with commodity wearable devices?* (Chapter 5)

RQ3. *What impact does context have on a gesture's design and a designer's gesture prototyping process?* (Chapter 6)

As a direct consequence of answering these questions, I made the following contributions to the space of motion gesture design research.

1. A triadic framework consisting of gestures, designers, and tools that provide a richer understanding of the types of gestures, considerations for and the process of gesture design, as well as factors that affect the design of gesture design tools.
2. The first ever explorations of *in situ* and *context-based prototyping* of motion gestures through the development of two generations of a smartphone-based tool, *Mogeste*, followed by *Comoge*.
3. A description of the challenges and advantages of designing motion gestures *in situ*, based on the first user study with both *professional* as well as student interaction designers.

7.2 Critical Reflection

Before I excitedly list my ideas for future work in the next section, I take a moment here to reflect on my work presented thus far. I have identified and discussed below two main criticisms of my work and provide my ideas for addressing the concerns in the future work.

7.2.1 Disentangling the Effects of the Study Design from the Tool Design

In the last chapter, I discussed several findings and issues uncovered during the summative evaluation of *Comoge*. While I did observe the study participants reflecting in the moment on these points, the questions that I asked during the semi-structured also prompted reflection. In this way, interviews let me probe deeper into both anticipated as well as unexpected findings. However, since the tool design informed the study design, it is difficult to say whether the questions asked during the interview or the tool itself was the cause of reflection. Moreover, the outdoor task's design was informed by the formative research including workshop in a course and interviews with experienced gesture designers and was meant to be representative of the tasks professional designers would be responsible for.

Upon reflection, I acknowledge limitations of the study design which could be addressed with future investigations. One way to disentangle the effects of study design and tool design is by conducting a comparative study between *Comoge* and a tool without support for context (for example, *Mogeste*) while keeping the tasks, interviews, and questionnaires constant. Another way is to treat *Comoge* as a technology probe to study its impact during longitudinal use by experienced gesture designers. The assumption here is that with the long-term use emergent behaviors could be captured and critical analysis of the tool's features could be facilitated. Finally, a workshop which provides an opportunity for semi-structured and collaborative tool use might elicit new insights and reveal limitations of the current iteration of the tool.

7.2.2 Augmenting the Machine Learning Back-End

As an astute reader might have noticed, throughout this document I haven't emphasized the machine learning back-end. This is because I designed the recognition system as informed by prior work done by researchers more knowledgeable than me in the domain of machine learning. For example, I chose a Support Vector Machine (SVM) classifier due to its good performance with a low number of training samples and encoded only standard statistical features which were found to carry high discriminative power [16]. Furthermore, I realized early that no recognition system can perform well for the large variety of potential gestures the users might explore unless manual effort is put into fine tuning the parameters every time a new gesture is considered. Hence, I decided to start with a reasonable system which, in my experience, over the past three years has served me well.

Despite my concerns about expending efforts on a better recognition system, I have thought of several ways to improve the recognition support in subsequent iterations of the tool. Firstly, I acknowledge the advantages a more adaptive recognition system could bring for the users of a gesture design tool. As discussed in detail below (see Section 7.3.4), a cloud-enabled back-end will allow experimentation with more sophisticated algorithms, collection of larger datasets, and on-the-fly learning of features that represent the data better than the fixed set of features *Comoge* currently employs. Secondly, integration of contextual information into the recognition system together with a robust context inferencing system will allow development of a context-based gesture recognition system, akin to LaViola's efforts [84]. For example, a tool could leverage the context tags such as mobility profile to train separate models under-the-hood and then select the right model during the test phase. Lastly, I can foresee an extension of the Everyday Gesture Library (EGL) idea that Ashbrook and colleagues [6, 75] had investigated. They proposed that samples for each new gesture be first queried against a large database of everyday motions for estimating the number of false positive triggers it may cause. In my version, the query could fetch information regarding potential contextual situations in which false positives might occur, this making a designer aware of the potential risks of implementing a gesture.

7.3 Open Challenges & Opportunities for Future Research

Many challenges and opportunities remain for developing our understanding of what gestures can and must be designed, how can designer's be made more effective in designing the right gestures, and, finally, how tools can support designers in designing and implementing better gestures faster. While the full coverage of a broad space these open-ended questions represent is left to future researchers, here, I will initiate the process by analyzing the assumptions that I implicitly make in my work in order to insinuate alternative paths.

In this thesis, I have assumed a narrative: that an *individual designer* is *designing & prototyping gestures* to be performed with a *single body part* at a time using a *standalone mobile tool* that leverages commodity and custom wearable sensing devices. Altering any of these four core assumptions yields an alternative narrative that suggests new avenues of research. I review each of these four avenues in order.

7.3.1 Collaborative Gesture Design

Most of the gesture authoring tools in literature, including the tool presented in this thesis, support the creative process of a single designer/developer. But as I have identified in Sections 3.3.3 & 4.4.1, there are opportunities for future tools to support collaboration within designers and between designers and other stakeholders. Furthermore, in both of these cases, the tools should strive to strengthen the role of a designer by empowering them with advanced capabilities that are complementary to their own strengths. *Collaboration within designers* is necessary for conceptualizing, prototyping, and documenting gestures that cross cultural boundaries. When such universal gestures are not feasible, a team of designers, each of which represents a different cultural context, can identify alternative gestures for conflicting scenarios without compromising on the usability, or recognizability.

Collaboration between designers and other stakeholders such as developers and end-users is crucial for ensuring proper implementation and use of designed gestures. Developers and designers already work together in industry settings, but often misunderstandings arise due to different objectives, expertise, and vocabulary. In my formative interviews, for example, designers reflected on their past experiences where their ideas were overruled by a developer's authoritative stance on the matters pertaining to recognizability of a gesture. Developers, on the other hand, remarked on the wild ideas that designers sometimes suggest.

7.3.2 Gesture-Driven Application Design

Prototyping motion gestures are the key objective of gesture authoring tools, but unless these gestures are implemented and used within gesture-driven application interfaces, motion gestures will never have an equal footing with other competing input methods such as multi-touch. Therefore, as noted in Section 4.2.2, future tools should broaden the scope to encompass all activities involved in the production of a gesture-based application. In this case, a designer should focus on end-to-end interface design process including visual design of screens and selection of input types and interactions. As a result, prototyping of new gestures is only undertaken if pre-existing gestures do not suffice or alternative input types are not appropriate.

However, the biggest hurdle is that, unlike ubiquitous support for touchscreen interactions, interface prototyping tools do not provide support for motion-based gestures, designers are not able to explore gestural input, even when it might be more suitable (e.g., microinteractions [7]). Unfortunately, if the application is not designed keeping 3-dimensional gestural input in mind from start the end-user experience will be suboptimal. For instance, a subtle visual feedback might not be noticeable when the device is in motion.

Another set of challenges arises due to what Norman and Nielsen [103] state as a lack of adherence to fundamental interaction design principles. According to them, although gestural interface introduces a sense of playfulness and freshness to UI design, they should still comply with following good usability practices suggested by following principles.

Feedback refers to the information provided to a user by the system in response to an action. For example, button press animation. A related but opposite concept is feedforward. Feedforward refers to providing advance information about available options when a user initiates an interaction [31].

Visibility and/or discoverability highlights the importance of making the controls discoverable through a systematic exploration of the interface.

Error recovery refers to the ability to recover from unintended input. Two common ways to support this are through an undo operation and also by seeking confirmation from the user when the input triggers a destructive action.

Consistency is about making and following user interface standards that persist across different applications on a platform. This allows an end-user to learn patterns and reduce effort as familiarity increases.

Reliability refers to both predictable responses to a user's input as well as prevention of accidental inputs.

Scalability argues for the appropriation of an interface to suit a form factor and available interaction techniques. For example, a checkbox for mouse input on a laptop is not suitable for touch input on a watch.

Additionally, *social acceptance* of gestures is another important dimension to consider while designing gestural interfaces. Montero et al. discuss the importance of understanding acceptance from the perspective of a gesture performer and also observers in a social setting [97]. To provide support for each of these principles will require a deeper investigation into existing interface design patterns, as well as explorations of new patterns pertaining to this input modality.

7.3.3 Multi-Device Motion Gestures

With the exception of few explorations, most tools support authoring of gestures with a single body part via a single sensing device. But in the light of recent developments (Section 4.2.1) which suggest each person already or in near future carrying multiple devices, I see an opportunity to develop tools that also support multi-device gestures. For example, a smartwatch worn on the left wrist and a smartphone held in the right hand can simultaneously move to define an interaction that signifies the transfer of information from the watch to the phone.

Below is a preliminary taxonomy of multi-device gestures.

- **Synchronous vs. sequential** Synchronous gestures are where both devices move simultaneously [68]. An example would be bringing up your arms in an X shape. Sequential gestures are where one device gesture is completed before the other device gesture begins. An example would be picking up your phone to your ear and then writing a number with your other hand to speed dial.
- **With or without physical contact** There are gestures which require the devices to make physical contact. One use is when your bump two phones together to transfer data between them [54].
- **Similar devices or different devices** A similar device gesture would be using two phones and tapping them together to share data. Using different devices like a watch and a phone to change views on a map where the phone can be used to move up, down, left or right, and the watch can be used to zoom in and out.
- **Multiple people vs. single person** The idea of multiples devices can be extended to multiple people, each holding a single device. For example, a multiple-person gesture would be two people holding two phones and tapping them together.

Besides expanding the vocabulary of gestures, multi-device gestures can also make gesture recognition more accurate. For instance, movement of one device can be used to disambiguate confusing or discard unintentional motions from the other device. However, a few technical challenges make their implementation difficult. First, synchronization of sensor data from multiple devices is problematic. Second, a large number of possible combinations of sensor streams even with two devices, makes an evaluation of the recognizers complicated.

7.3.4 Other Manifestations of a Tool

In contrast to prior work that focused on desktop-based tools, I only explored standalone smartphone-based tools with an emphasis on mobility and in situ design. Given the powerful computing abilities of the current generation smartphones, I was able to successfully prove usability, and benefits of tools developed for them. However, in retrospection, I believe relaxing this constraint a little bit will provide support for even newer possibilities while maintaining mobility. To this end, I consider cloud-enabled backend support and wearable-based versions in future.

For the *cloud-enabled* version, I envision replacing local storage and processing of sensor data with more powerful and infinitely scalable cloud-based storage and computing solutions. As I see, this shift will provide following advantages.

- Enable remote collaborations on gesture designs.
- Allow experimentation with large datasets and more advanced pattern recognition algorithms.
- Support shift towards real-time feedback (for example, on false-positive characterization) and guidance.
- Increase flexibility in number and form-factor of the front-end clients.

As for the *standalone wearable* version, With or without the cloud, I visualize a setup where small custom wearable sensors can directly relay data to a nearby wearable device like a smartwatch, or a head-mounted Google Glass. Through a minimal interface, recording, labeling, and testing of the motion gestures can be easily realized. The biggest advantages of this setup, in comparison to the setup used in this thesis, are increased mobility and naturalness of gesture design. Finally, wearable tools will promote a paradigm of continuous testing and refinement of motion gestures.

7.4 *Proposed Guidelines for Future Systems*

Based on the formative interviews and two summative evaluations with professional and student designers, I identified following needs that motion gesture prototyping tools should fulfill.

7.4.1 Avoid Creative Block

A gesture design tool should provide means to overcome a designer's creative block. First, by providing examples of existing gesture designs and allowing a designer to build onto them, a tool can seed creativity. These examples could be preloaded, designer's own prior work, or designs shared by other designers. Second, most of my participants had trouble 'thinking in the moment' and coming up with gesture designs. Because, creativity is serendipitous, the ubiquitous availability of the tool will allow the designer to work anytime, anyplace, and quickly evaluate their designs.

7.4.2 Refocus Effort to Design

From the user study, I realized that gesture design and gesture prototyping are not independent of each other but are co-dependent and co-occur. By allowing a designer to try different designs with little effort (few examples per gesture), a gesture design tool can help her focus on deeper questions about what's possible, what makes sense to the task and end-users, how can she evaluate her designs, etc. A tool can also help by making documentation of the design process easy. This is particularly important during early-stage prototyping when she is working on multiple design alternatives, making recollection of details difficult. Moreover, she may want to quickly annotate designs with observations made during an in-context evaluation. By managing tasks of documentation, tweaking machine learning parameters, and programming that are pertinent to prototyping, a prototyping tool can help refocus resources to gesture design.

7.4.3 Support Context-Aware Design

Selection of appropriate motion gestures depends on several types of contextual information including culture, social situation, activity, application, etc. For example, one study participant wanted to record who performed a gesture. Her rationale was that "children might perform gesture like this but an adult will perform gesture like that, maybe it's a little different." Thus, in addition to facilitating in situ design, motion-gesture prototyping tools should support capture and access of relevant contextual information. Moreover, it should allow a designer to evaluate their gesture designs within that context and take note of contextual references.

7.4.4 Explore Alternative (Interaction) Modalities to Use the Tool

As discussed before, there is a requirement for alternative modalities for input-feedback-output loop other than screen-based visual modality or touch gestures. This holds true specifically while recording the gesture when either the display is not available or when reaching the visual display to provide input would alter the task. Therefore, other modalities like speech input and output along with audio or haptic feedback should also be considered in addition to visual elements.

7.4.5 Support Multiple Methods of Analysis and Reflection

A tool is also a means of reflection. Its usage raises new questions in a designer's mind. One way a tool concretely supports this practice is by allowing review and edit. However, what type of recognizer feedback to expose, and what types of views to provide are open questions. My study participants suggested the inclusion of intra-class comparison values, test logs, timestamps for recorded samples, etc. Other times a tool itself could be the object of reflection. For instance, one designer's model of my tool was that of a voice recorder.

7.5 Conclusion

Motion gestures have the potential of creating embodied interactions that delight us, exercise our bodies and minds, and prove natural where other popular interaction techniques fail. However, unless we empower interaction designers, those who are (or will be) responsible for envisioning, prototyping, evaluating, and ultimately implementing these motion gestures, motion gestures don't stand a chance. Moreover, in contrast with previous work, through this thesis, I argue that until we liberate the gesture design process and designers from the confines of a laboratory and facilitate design in naturalistic contexts, designed gestures will seldom be appropriate for a wide variety of contexts their users inhabit. To this end, I presented two iterations of a smartphone-based tool that facilitates rapid, in situ design of motion gestures that are informed by important contextual factors. Together, with the guidelines and new research avenues listed above, this work provides a foundation for future researchers and developers of motion gesture design tools.

APPENDIX A

STUDY INSTRUMENTS

A.1 Elicitation Study with First Year HCI Students

A.1.1 Task Sets

SET 1

Set 1	**You have to call your team-mate to discuss the next assignment.**
Tasks:	1) Place the call. 2) You have discussed all the specifics of the assignment and now have to hang-up. End the call.

SET 2

Set 2	**You have an incoming call from your friend.**
Tasks:	1) Answer the call. 2) You are sitting in a class and can't take a call. Ignore the call.

SET 3

Set 3		
Tasks:	1)	You want quickly search the open time of an eatery. Initiate Voice search
	2)	You want to open the Gallery to view photos. Select the Gallery app.

SET 4

Set 4:	**You are exploring the map of your neighborhood.**	
Tasks:	1)	Zoom-in to view the exact details.
	2)	Zoom out to see the overview
	3)	Pan right on the map to view the content
	4)	Pan left on the map to view the content
	5)	Pan up on the map to view the content
	6)	Pan down on the map to view the content

SET 5

Set 5:	**You are reviewing today's weather to see if it will rain today or not on a weather app.**
Tasks:	If pages are arranged horizontally: 1) Scroll to the next page (horizontally) 2) Scroll to the previous page (horizontally) If pages are arranged vertically: 3) Scroll to the next page (vertically) 4) Scroll to the previous page (vertically)

SET 6

Set 6:	**You have Calendar app open but wish to take a quick note in Evernote**
Tasks:	1) Switch to the next app 2) Switch to previous app 3) Open the Home screen

A.1.2 Worksheet

In-class activity to "Design and Evaluate motion gestures for commercial wearable devices"

Team Number: ________ Gesture Set: __________ Activity Number: _______ Team Members (Initials): __________

1. Design Motion gestures for all the tasks mentioned in the gesture set. Do not worry about how the gesture would be detected or implemented. Let the creativity flow!
2. For each task, video record the gestures you have designed. Use your phones to record the video

Task No.	Short description on how to perform the gesture.	Rationale on why you chose/design the gesture that you did.

In-class activity to “Design and Evaluate motion gestures for commercial wearable devices”

Task No.	Short description on how to perform the gesture.	Rationale on why you chose/design the gesture that you did.

In-class activity to "Design and Evaluate motion gestures for commercial wearable devices"

3. For each of the tasks in the set:
   a. Perform the designed gesture 4-5 times.
   b. Answer the following as a team. If your team doesn't have consensus, give reason why?

   Task Number: _______________

	Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree
The gesture designed matches its intended use.					

	Very Difficult	Difficult	Neutral	Easy	Very Easy
Performing the gesture would be:					
Remembering the gesture would be:					

	Never	Almost never	Occasionally	Almost every time	Frequently use
The gesture will be used:					

What you would think if you saw someone else performing this gesture when:

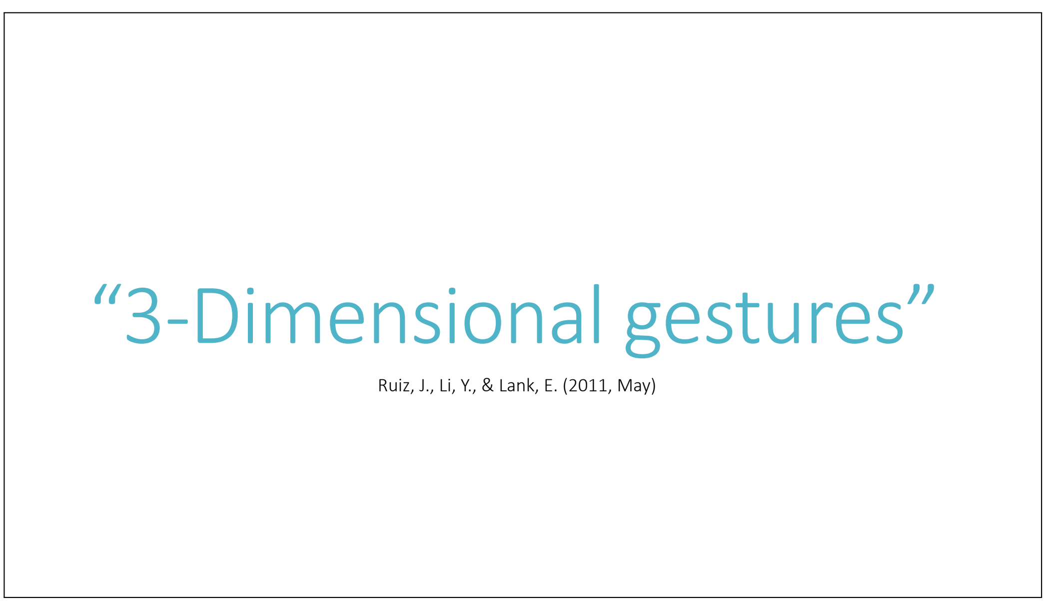
	Absolutely Inappropriate	Inappropriate	Neutral	Appropriate	Absolutely Appropriate
Walking down the street.					
Eating in a restaurant with friend					
Sitting in a public transport					
At home					

From scale 1 (Embarrassed) to 6 (Comfortable), How would you feel performing this gesture in the following situations?

	Embarrassed	Slightly Embarrassed	Neutral	Just Fine	Comfortable
Walking down the street.					
Eating in a restaurant with friends					
Sitting in a public transport					
at home					

A.1.3 Slides

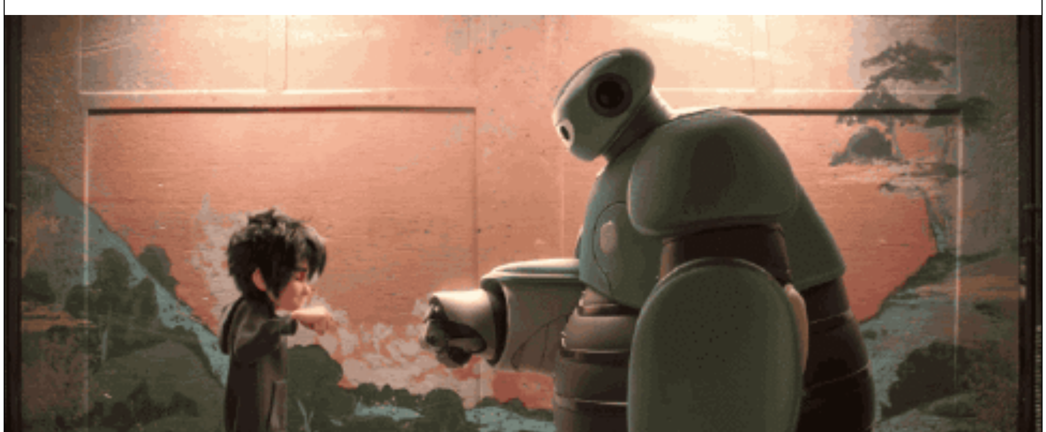

Motion Gestures
Apurva Gupta, MS-HCI
Aman Parnami, PhD-HCC


“3-Dimensional gestures”
Ruiz, J., Li, Y., & Lank, E. (2011, May)

Waving

Karate Chop

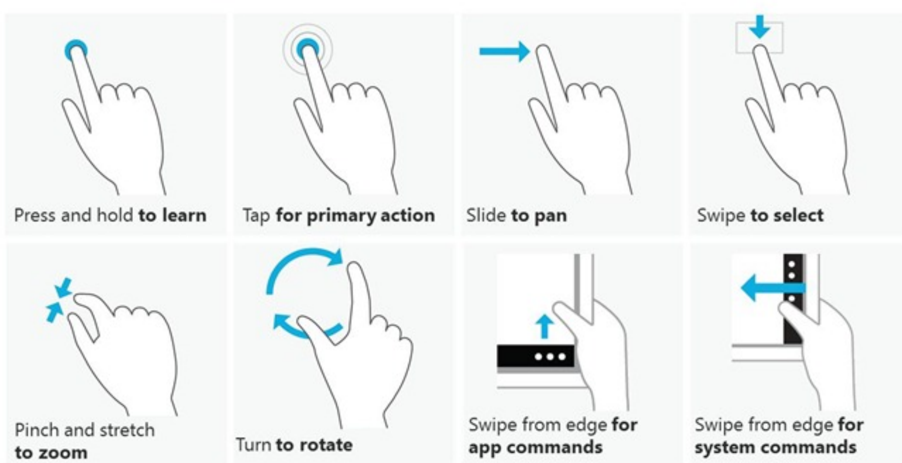
Kung-fu ???



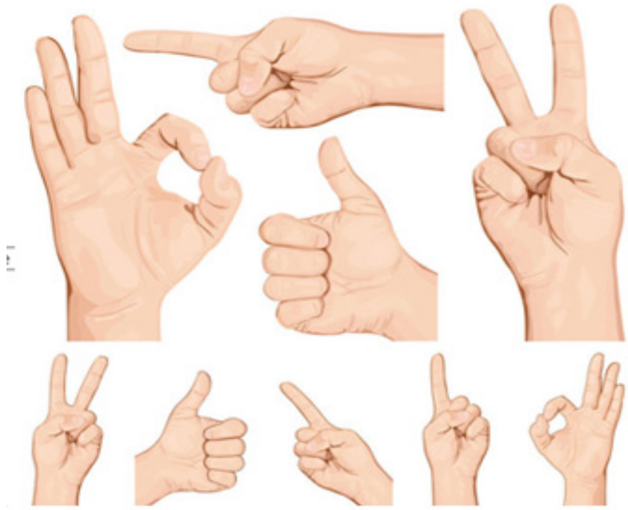
Bla-la-la-la-la-la-la

“2-Dimensional gestures”

Ruiz, J., Li, Y., & Lank, E. (2011, May)



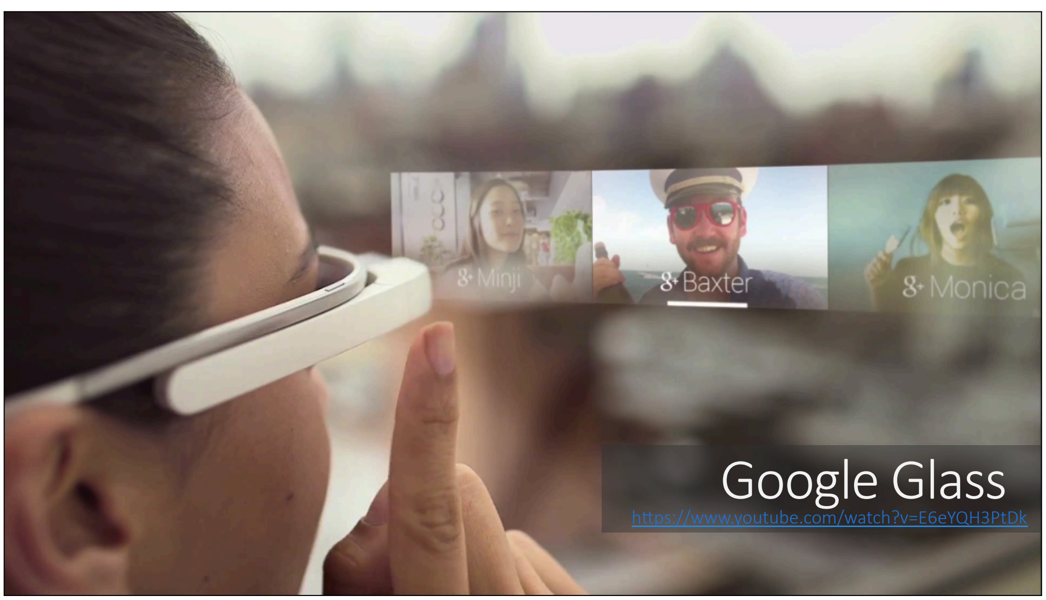
Surface Gestures



Static Gestures

Motion Gestures in Commercial Wearable Devices


Merchandise
Moto 360
https://www.youtube.com/watch?v=J7kniJ4ovKc


Minji
Baxter
Monica
Google Glass
https://www.youtube.com/watch?v=E6eYQH3PtDk

Designing Motion Gestures

Part 1

Brief!

- Group of TWO!!
- Wrist-mounted Group and Head-mounted Group!!
- Design Gestures!!
- Answer few questions!!
- Follow-up HW Assignment…

Evaluating Motion Gestures

Part 2

A.2 Professional Interviews

A.2.1 Interview Questions

Date_*October 16th 2015*_ **Volunteer #**_______

INTERVIEW GUIDE

Georgia Institute of Technology

Project Title: **Mobile Tool for Designing and Testing New Input Techniques that leverage Commodity Sensors**

INTRODUCTION

Hi! I am Aman. I am a third year PhD student in the HCI program at Georgia Tech. I am interested in building tools for designers who prototype devices and interactions. I am here to talk to you specifically about your experience with designing/developing gestural interaction.

[Ask if you can record the interview. Tell them that they can stop the recording anytime they want to go off record.]

BACKGROUND

- Are you familiar with the wrist rotation gesture on Android smart watches? If so, what do you think about it? If not, [show how the gesture works]
- Have you designed gestures yourself? If yes, walk me through an example.

OR

- If no, let's consider a scenario in which taking out your phone is not appropriate, time consuming, or just plain difficult. Let's think about how you could operate your phone using gestures with a watch or Glass. Walk me through the steps you would take to prototype these gestures.

UNDERSTANDING THE GESTURE DESIGN PROCESS

Planning and requirement gathering

- What was the goal of the activity?
- Whom were you designing for?
- How did you generate insights about what gestures will work? Did you observe/ask people? Envisioned gestures?
- How did you conceptualize gestures? Sketching, enacting, textual description, etc.
- Did you work alone or with others? What was your role? If team, what skills do others have?

Prototyping

- What gestures did you come up with?
- How did you prototype those gestures? [ask about process] [ask about tools]
- What was your setup like?
- Can you recall an occasion on which you thought, "I wish there was a way to do X"?

Evaluation

- How did you test the gestures? Did you recruit others for testing? If so, whom?
- Did you go out in the field?
- How did you make sure the gestures you created would not be confused with some other gesture?
  - What were the common sources of confusion for your gestures?

Post-prototyping process

- What was the next step after you evaluated the gestures? [ask about role of developers at this stage]

Date_*October 16th 2015*_ **Volunteer #**_______

MACHINE LEARNING EXPERIENCE

- What is your exposure to Machine Learning/Pattern Recognition? Do you keep it in mind while designing gestures?

THOUGHTS ON SPECIFIC FEATURES OF TOOLS

Representation

- If you were involved with creation of several gestures, how would you help yourself remember a gesture?
    - Did you come up with names for your gestures?
- What would allow you to repeat a gesture with confidence?
- Would you mind being video recorded while performing a gesture for your own review later?

Recognizer Feedback

- In what ways can a tool help you design reliable gestures?
- What type of feedback would you want?
- What if you were designing multiple gestures for the same application? Would you want the system to behave differently?
- Can you recall an occasion on which you had to make a change based on feedback from a tool? What did you change?

FEEDBACK ON ANY TOOL THEY HAVE USED

- How robust does the tool has to be to be a benefit? Is development cost a concern?

[Ask if the participant has any last thoughts or questions for you.]

[Thank the participant.]

FOLLOW UP

If you are you interested in testing the tool, I will be happy to provide you a copy.

A.3 Mogeste User Study with 7 Novice Designers

A.3.1 Interview Questions

Date: Participant No: Conducted By:

INTERVIEW GUIDE

Georgia Institute of Technology

Designing an in-context Motion Gesture Prototyping tool for designers

Introduction

Hi! Thanks for taking out time to participate in the study. The project is on designing a prototyping tool for motion gestures for designers. The study will be conducted in 2 parts. Part one will involve an introduction to the motion gestures, a design task and some Follow-up questions based on the task.

In the second part, I will introduce you to the tool and give you a task for that. There will be some follow-up question to that.

The participation in the study is voluntary and you can leave at any point in time. There are no risks associated with the study and all there will not be any compensation.

Is it fine to video and audio record the session? You can ask to stop the recording anytime.

Background [getting them warmed up]

1. Are you familiar with the wrist rotating gesture on Android smart watch? If so, what do you think about it? If not, [show how the gesture works]
2. Have you designed gestures yourself? If yes, walk me through an example.
3. If no, [Some example]: You are walking in the dark and holding a bag in one hand and phone on the other. You shake the phone to switch the flashlight on or you are in an emergency and make "E" in air to call 911.

PART 1

Design task

Design 3 different gestures for each of the given task using the hand in which you are holding the phone.

Assumptions: Do not worry about recognition and assume the proceeding steps have already been taken/done. Recognition is happening on the hand holding the phone. Also, try to think out loud about your thoughts and process.

1) You have to call your team-mate to discuss the next assignment.

Place the call.

2) You have discussed all the specifics of the assignment and now have to hang-up.

End the call.

Date: Participant No: Conducted By:

Interview Questions

1. **Design**
    a. What gestures did you come up with? Why?
    b. How did you generate insights about what gestures will work?
    c. What additional information would you need to design the gesture?
    d. Prioritize the Gestures. Why?
2. **Prototyping**
    a. If you were to prototype the gesture, how would you do it?
    b. What do you think you would need to prototype the gestures?
    c. [hardware], software, what do sensors connect to, what data are you gathering, how would you use the data]
    d. Would you work alone or with others? If other, Whom? Why? What skills do others have?
    e. What would your role be?
3. **Evaluation**
    a. How would you evaluate the gestures you just designed for reliability?
    b. How would you make sure the gestures you designed would not be confused with some other natural movements/gesture throughout the day?
    c. How would you evaluate if the gestures work for other users?
    d. What other questions would you seek answers to? ____
    e. How confident are you about the gestures you have designed?
4. **Misc.**
    a. If you were to hand this over to developer, what according to you would be the best way to package everything?
    b. What is your exposure to Machine Learning/Pattern Recognition? Did you keep it in mind while designing gestures?
    c. If there were a tool that would help you design the gestures better, what should some of its features be?

A.3.2 Worksheet

"Designing an in-context Motion Gesture Prototyping tool for designers" Participant No: Date:

Design 3 different gestures for each of the given task using the hand in which you are holding the phone.

1. You have to call your team-mate to discuss the next assignment.
   Place the call.
2. You have discussed all the specifics of the assignment and now have to hang-up.
   End the call.

Design	Short description on how to perform the gesture. [Place the call]	Short description on how to perform the gesture. [End the call]

"Designing an in-context Motion Gesture Prototyping tool for designers" Participant No: Date:

Design	Short description on how to perform the gesture. [Place the call]	Short description on how to perform the gesture. [End the call]

A.4 Comoge User Study with 12 Students, 5 Professionals

A.4.1 Interview Questions

QUESTIONS

- How would you describe this tool to one of your classmates?
- What was your experience with the tool? (What is good and bad?)
  - Do you think the tool could be useful? How? Whom?
  - What should be added to the tool to make it more useful to you?
  - Do you think this tool would make it easier to design gestures than other ways to design gestures?
    - Do you think you could do different things with this tool than without it?
  - How do you think this tool might impact your work as a designer with developers in the same team? How?
- What do you think about the gestures you designed and prototyped with the tool?
  - Do you think the gestures you designed are easy to perform? Why? Why not?
  - Do you think the gestures you designed are socially acceptable? Why? Why not?
  - Do you think any of the gestures you designed will be confused with some other natural movements occurring throughout the day? By recognizer? By other people?
  - Did your gestures change during prototyping? In what ways? Why?
- Compare and contrast your experiences of designing gestures indoors and outdoors?
  - How were they similar?
  - How were they different?
  - Did adding examples for each case help?

A.4.2 Introduction Text

BOTH RESEARCHER AND PARTICIPANT SHOULD SILENCE THEIR MOBILE DEVICES AND PUT THEM AWAY.

Let's start.

What are motion gestures?
Motion gestures are intentional, voluntary 3D movements of body parts detected through body-worn or handheld inertial sensors.

What is Comoge?
Tool which helps designers like you design and test motion gestures for various wearable/mobile devices in different activity contexts such as walking, in vehicle, etc.

It senses data from the inertial sensors and allows the user to test the efficiency of gesture in terms of recognition. I.e., whether the gesture in discussion would be detected or not? Will there be false-positives? Etc.

Overall, the tool is designed to help users make better and informed decisions while designing motion gestures.

A.4.3 Questionnaire

System Usability Scale

Please tick an option to mark your response. Mark the center option if a question is not applicable.

* Required

1. **Participant ID** *

2. **I think that I would like to use this system frequently**
Mark only one oval.

	1	2	3	4	5	
Strongly disagree	◯	◯	◯	◯	◯	Strongly agree

3. **I found the system unnecessarily complex**
Mark only one oval.

	1	2	3	4	5	
Strongly Disagree	◯	◯	◯	◯	◯	Strongly Agree

4. **I thought the system was easy to use**
Mark only one oval.

	1	2	3	4	5	
Strongly Disagree	◯	◯	◯	◯	◯	Strongly Agree

5. **I think that I would need the support of a technical person to be able to use this system**
Mark only one oval.

	1	2	3	4	5	
Strongly Disagree	◯	◯	◯	◯	◯	Strongly Agree

6. **I found the various functions in this system were well integrated**
Mark only one oval.

	1	2	3	4	5	
Strongly Disagree	◯	◯	◯	◯	◯	Strongly Agree

7. **I thought there was too much inconsistency in this system**
Mark only one oval.

	1	2	3	4	5	
Strongly Disagree	◯	◯	◯	◯	◯	Strongly Agree

8. **I would imagine that most people would learn to use this system very quickly**
Mark only one oval.

	1	2	3	4	5	
Strongly Disagree	◯	◯	◯	◯	◯	Strongly Agree

9. **I found the system very cumbersome to use**
Mark only one oval.

	1	2	3	4	5	
Strongly Disagree	◯	◯	◯	◯	◯	Strongly Agree

10. **I felt very confident using the system**
Mark only one oval.

	1	2	3	4	5	
Strongly Disagree	◯	◯	◯	◯	◯	Strongly Agree

11. **I needed to learn a lot of things before I could get going with this system**
Mark only one oval.

	1	2	3	4	5	
Strongly Disagree	◯	◯	◯	◯	◯	Strongly Agree

Questionnaire for User Interaction Satisfaction

For each of the following questions, fill in 0-9 or leave blank if question is not applicable.

Overall Reactions To The Software

12. *Mark only one oval.*

	0	1	2	3	4	5	6	7	8	9	
terrible	◯	◯	◯	◯	◯	◯	◯	◯	◯	◯	wonderful

13. *Mark only one oval.*

	0	1	2	3	4	5	6	7	8	9	
difficult	⬭	⬭	⬭	⬭	⬭	⬭	⬭	⬭	⬭	⬭	easy

14. *Mark only one oval.*

	0	1	2	3	4	5	6	7	8	9	
frustrating	⬭	⬭	⬭	⬭	⬭	⬭	⬭	⬭	⬭	⬭	satisfying

15. *Mark only one oval.*

	0	1	2	3	4	5	6	7	8	9	
dull	⬭	⬭	⬭	⬭	⬭	⬭	⬭	⬭	⬭	⬭	stimulating

16. *Mark only one oval.*

	0	1	2	3	4	5	6	7	8	9	
rigid	⬭	⬭	⬭	⬭	⬭	⬭	⬭	⬭	⬭	⬭	flexible

Skip to question 17.

Screen

17. **Characters on the computer screen**
Mark only one oval.

	0	1	2	3	4	5	6	7	8	9	
hard to read	⬭	⬭	⬭	⬭	⬭	⬭	⬭	⬭	⬭	⬭	easy to read

18. **Organization of information on screen**
Mark only one oval.

	0	1	2	3	4	5	6	7	8	9	
confusing	⬭	⬭	⬭	⬭	⬭	⬭	⬭	⬭	⬭	⬭	very clear

19. **Sequence of screens**
Mark only one oval.

	0	1	2	3	4	5	6	7	8	9	
confusing	⬭	⬭	⬭	⬭	⬭	⬭	⬭	⬭	⬭	⬭	very clear

Terminology And System Information

20. **Use of terms throughout system**
Mark only one oval.

	0	1	2	3	4	5	6	7	8	9	
inconsistent											consistent

21. **Computer terminology is related to the task you are doing**
Mark only one oval.

	0	1	2	3	4	5	6	7	8	9	
never											always

22. **Position of messages on the screen**
Mark only one oval.

	0	1	2	3	4	5	6	7	8	9	
inconsistent											consistent

23. **Messages on screen which prompt user for input**
Mark only one oval.

	0	1	2	3	4	5	6	7	8	9	
confusing											clear

24. **Computer keeps you informed about what it is doing**
Mark only one oval.

	0	1	2	3	4	5	6	7	8	9	
never											always

25. **Error messages**
Mark only one oval.

	0	1	2	3	4	5	6	7	8	9	
unhelpful											helpful

Learning

26. **Learning to operate the system**
Mark only one oval.

	0	1	2	3	4	5	6	7	8	9	
difficult											easy

27. **Exploring new features by trial and error**
Mark only one oval.

	0	1	2	3	4	5	6	7	8	9	
difficult											easy

28. **Remembering names and use of commands**
Mark only one oval.

	0	1	2	3	4	5	6	7	8	9	
difficult											easy

29. **Tasks can be performed in a straightforward manner**
Mark only one oval.

	0	1	2	3	4	5	6	7	8	9	
never											always

30. **Help messages on the screen**
Mark only one oval.

	0	1	2	3	4	5	6	7	8	9	
unhelpful											helpful

System Capabilities

31. **System speed**
Mark only one oval.

	0	1	2	3	4	5	6	7	8	9	
too slow											fast enough

32. **System reliability**
Mark only one oval.

	0	1	2	3	4	5	6	7	8	9	
unrealiable											reliable

33. **System tends to be**
Mark only one oval.

	0	1	2	3	4	5	6	7	8	9	
noisy											quiet

34. **Correcting your mistakes**
Mark only one oval.

	0	1	2	3	4	5	6	7	8	9	
difficult											easy

35. **Experienced and inexperienced users' needs are taken into consideration**
Mark only one oval.

	0	1	2	3	4	5	6	7	8	9	
never											always

Usability And UI

36. **Use of colors and sounds**
Mark only one oval.

	0	1	2	3	4	5	6	7	8	9	
poor											good

37. **System feedback**
Mark only one oval.

	0	1	2	3	4	5	6	7	8	9	
poor											good

38. **System response to errors**
Mark only one oval.

	0	1	2	3	4	5	6	7	8	9	
awkward											gracious

39. **System messages and reports**
Mark only one oval.

	0	1	2	3	4	5	6	7	8	9	
poor	⬭	⬭	⬭	⬭	⬭	⬭	⬭	⬭	⬭	⬭	good

40. **System clutter and UI "noise"**
Mark only one oval.

	0	1	2	3	4	5	6	7	8	9	
poor	⬭	⬭	⬭	⬭	⬭	⬭	⬭	⬭	⬭	⬭	good

Exit Survey

41. **What is your gender?**
Mark only one oval.

⬭ Male
⬭ Female
⬭ Other: ____________

42. **Which is your dominant hand?**
Mark only one oval.

⬭ Right Hand
⬭ Left Hand
⬭ Ambidextrous (Either hand can serve as the dominant hand)

43. **What is your experience working with sensors?**
Mark only one oval.

	1	2	3	4	5	
Novice	⬭	⬭	⬭	⬭	⬭	Expert

44. **What is your experience with programming?**
Mark only one oval.

	1	2	3	4	5	
Novice	⬭	⬭	⬭	⬭	⬭	Expert

45. **What is your experience with Interaction Design? (Interaction design, often abbreviated as IxD, is defined as "the practice of designing interactive digital products, environments, systems, and services.")**
Mark only one oval.

	1	2	3	4	5	
Novice	⬭	⬭	⬭	⬭	⬭	Expert

46. **What is your experience with Machine Learning/Pattern Recognition?**
Mark only one oval.

	1	2	3	4	5	
Novice	⬭	⬭	⬭	⬭	⬭	Expert

47. **Have your ever used 3D motion gestures to interact with a computer application ?**
Mark only one oval.

- Yes
- No

48. **Have your ever designed new 3D motion gestures for interacting with a computer application ?**
Mark only one oval.

- Yes
- No

49. **Which one of the following devices have you used before?**
Check all that apply.

- [ ] Smart watch (e.g., Apple watch)
- [ ] Smart phone
- [ ] Smart head mounted device (e.g., Google Glass)

50. **Which one of the following devices you own?**
Check all that apply.

- [ ] Smart watch (e.g., Apple watch)
- [ ] Smart phone
- [ ] Smart head mounted device (e.g., Google Glass)

A.4.4 Task Sheets

Task 1(Indoors)

Your task is to design **smartwatch** gestures for an indoor combat game involving evil fruits that are attacking you.

STEPS

1. Create a project titled "Combat with fruits"
2. Create following three gestures within this project
    a. Shoot a Melon
    b. Slice a Pineapple In Air
    c. Smash a Coconut On Table
3. Record at least three examples of each gesture
4. Perform at least three tests

Instructions

- Think out loud what you are doing
- Ask the researcher if you have any questions before, during, or after the task.

Task 2 (Outdoors)

Suppose you are listening to your favorite music playing on your phone, but find it difficult to operate it through the touchscreen. Your task is to design gestures for operating a music player on the go.

REQUIREMENTS

1. Identify 4 common operations for a music player controller while on the go
2. For smartwatch: Design and prototype at least one smartwatch-based motion gesture for each operation
3. For Google Glass: Design and prototype at least one Glass-based motion gesture for each operation
4. Record at least three examples of each gesture for two activity contexts: 1) while on foot, 2) while on vehicle
5. Perform tests to make sure gestures for both devices work in both activity contexts

DESIGN CRITERIA

- Gestures should be easy to perform
- Gestures should be socially acceptable
- You should maintain a high goodness score for all gestures
- Gestures should work while on foot (walking/running), and while on vehicle (bus)
- Consider operating your music player on phone with two different devices: a smartwatch, Google Glass

Instructions

- Researcher will be accessible, but you are supposed to manage on your own.
- If the app crashes, then restart it and resume whatever you were doing.

REFERENCES

[1] Abowd, G. D., "What next, ubicomp?: celebrating an intellectual disappearing act," in *Proc. Ubicomp*, pp. 31–40, 2012.

[2] Abowd, G. D. and Mynatt, E. D., "Charting past, present, and future research in ubiquitous computing," *ACM TOCHI*, vol. 7, no. 1, pp. 29–58, 2000.

[3] Amini, S. and Li, Y., "CrowdLearner: rapidly creating mobile recognizers using crowdsourcing," in *Proc. CHI*, pp. 163–172, 2013.

[4] Amma, C., Georgi, M., and Schultz, T., "Airwriting: A Wearable Handwriting Recognition System," *Personal Ubiquitous Comput.*, vol. 18, no. 1, pp. 191–203, 2014.

[5] Arefin Shimon, S. S., Lutton, C., Xu, Z., Morrison-Smith, S., Boucher, C., and Ruiz, J., "Exploring Non-touchscreen Gestures for Smartwatches," in *Proc. CHI*, pp. 3822–3833, ACM, 2016.

[6] Ashbrook, D. and Starner, T., "MAGIC: A Motion Gesture Design Tool," in *Proc. CHI*, pp. 2159–2168, 2010.

[7] Ashbrook, D. L., *Enabling mobile microinteractions.* PhD thesis, Georgia Institute of Technology, Atlanta, GA, 2010.

[8] Bayta, M. A., Yemez, Y., and zcan, O., "Hotspotizer: End-user Authoring of Mid-air Gestural Interactions," in *Proc. NordCHI*, pp. 677–686, ACM, 2014.

[9] Beaudouin-Lafon, M., "Instrumental interaction: an interaction model for designing post-WIMP user interfaces," in *Proc. CHI*, pp. 446–453, 2000.

[10] Bedri, A., Verlekar, A., Thomaz, E., Avva, V., and Starner, T., "Detecting Mastication: A Wearable Approach," in *Proc. ICMI*, pp. 247–250, ACM, 2015.

[11] Bellotti, V., Back, M., Edwards, W. K., Grinter, R. E., Henderson, A., and Lopes, C., "Making Sense of Sensing Systems: Five Questions for Designers and Researchers," in *Proc. CHI*, pp. 415–422, 2002.

[12] Benford, S., Schndelbach, H., Koleva, B., Anastasi, R., Greenhalgh, C., Rodden, T., Green, J., Ghali, A., Pridmore, T., Gaver, B., Boucher, A., Walker, B., Pennington, S., Schmidt, A., Gellersen, H., and Steed, A., "Expected, Sensed, and Desired: A Framework for Designing Sensing-based Interaction," *ACM TOCHI*, vol. 12, pp. 3–30, Mar. 2005.

[13] Bolt, R. A., ""Put-that-there": Voice and Gesture at the Graphics Interface," in *Proceedings of the 7th Annual Conference on Computer Graphics and Interactive Techniques*, pp. 262–270, ACM, 1980.

[14] Bragdon, A., Zeleznik, R., Williamson, B., Miller, T., and LaViola, Jr., J. J., "GestureBar: Improving the Approachability of Gesture-based Interfaces," in *Proc. CHI*, pp. 2269–2278, ACM, 2009.

[15] Brooke, J. and others, "SUS-A quick and dirty usability scale," *Usability evaluation in industry*, vol. 189, no. 194, pp. 4–7, 1996.

[16] Bulling, A., Blanke, U., and Schiele, B., "A tutorial on human activity recognition using body-worn inertial sensors," *CSUR*, vol. 46, no. 3, p. 33, 2014.

[17] Burges, C. J., "A tutorial on support vector machines for pattern recognition," *Data mining and knowledge discovery*, vol. 2, no. 2, pp. 121–167, 1998.

[18] Buxton, W., "Chunking and phrasing and the design of human-computer dialogues," in *IFIP Congress*, pp. 475–480, 1986.

[19] Buxton, W., *Sketching user experiences getting the design right and the right design.* Amsterdam; Boston: Elsevier/Morgan Kaufmann, 2007.

[20] Card, S. K., Mackinlay, J. D., and Robertson, G. G., "The design space of input devices," in *Proc. CHI*, pp. 117–124, 1990.

[21] Chan, E., Seyed, T., Stuerzlinger, W., Yang, X.-D., and Maurer, F., "User Elicitation on Single-hand Microgestures," in *Proc. CHI*, pp. 3403–3414, ACM, 2016.

[22] Chan, L., Chen, Y.-L., Hsieh, C.-H., Liang, R.-H., and Chen, B.-Y., "CyclopsRing: Enabling Whole-Hand and Context-Aware Interactions Through a Fisheye Ring," in *Proc. UIST*, UIST '15, (New York, NY, USA), pp. 549–556, ACM, 2015.

[23] Chan, L., Hsieh, C.-H., Chen, Y.-L., Yang, S., Huang, D.-Y., Liang, R.-H., and Chen, B.-Y., "Cyclops: Wearable and Single-Piece Full-Body Gesture Input Devices," in *Proc. CHI*, pp. 3001–3009, ACM, 2015.

[24] Chang, C.-C. and Lin, C.-J., "LIBSVM: A Library for Support Vector Machines," *TIST*, vol. 2, no. 3, pp. 27:1–27:27, 2011.

[25] Chen, X. A., Schwarz, J., Harrison, C., Mankoff, J., and Hudson, S. E., "Air+Touch: Interweaving Touch & In-air Gestures," in *Proc. UIST*, pp. 519–525, 2014.

[26] Chin, J. P., Diehl, V. A., and Norman, K. L., "Development of an Instrument Measuring User Satisfaction of the Human-computer Interface," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '88, (New York, NY, USA), pp. 213–218, ACM, 1988.

[27] Cohn, G., Gupta, S., Lee, T.-J., Morris, D., Smith, J. R., Reynolds, M. S., Tan, D. S., and Patel, S. N., "An ultra-low-power human body motion sensor using static electric field sensing," in *Proc. Ubicomp*, pp. 99–102, 2012.

[28] Cohn, G., Morris, D., Patel, S., and Tan, D., "Humantenna: using the body as an antenna for real-time whole-body interaction," in *Proc. CHI*, pp. 1901–1910, 2012.

[29] Crossan, A., Brewster, S., and Ng, A., "Foot Tapping for Mobile Interaction," in *Proc. BCS*, pp. 418–422, British Computer Society, 2010.

[30] Cypher, A. and Halbert, D. C., *Watch what I do: programming by demonstration.* MIT press, 1993.

[31] Delamare, W., Coutrix, C., and Nigay, L., "Designing Guiding Systems for Gesture-based Interaction," in *Proc. EICS*, pp. 44–53, ACM, 2015.

[32] Dey, A. K., "Understanding and Using Context," *Personal Ubiquitous Comput.*, vol. 5, pp. 4–7, Jan. 2001.

[33] Dey, A. K., Abowd, G. D., and Salber, D., "A conceptual framework and a toolkit for supporting the rapid prototyping of context-aware applications," *Human-computer interaction*, vol. 16, no. 2, pp. 97–166, 2001.

[34] Dey, A. K., Hamid, R., Beckmann, C., Li, I., and Hsu, D., "a CAPpella: programming by demonstration of context-aware applications," in *Proc. CHI*, pp. 33–40, 2004.

[35] Dey, A. K., Sohn, T., Streng, S., and Kodama, J., "iCAP: Interactive prototyping of context-aware applications," in *Pervasive Computing*, pp. 254–271, Springer, 2006.

[36] Domingos, P., "A Few Useful Things to Know About Machine Learning," *Commun. ACM*, vol. 55, pp. 78–87, Oct. 2012.

[37] Dourish, P., *Where the action is: the foundations of embodied interaction.* MIT press, 2004.

[38] Fails, J. and Olsen, D., "A design tool for camera-based interaction," in *Proc. CHI*, pp. 449–456, 2003.

[39] Fails, J. A. and Olsen, Jr., D. R., "Interactive Machine Learning," in *Proc. IUI*, pp. 39–45, ACM, 2003.

[40] Fiebrink, R., Trueman, D., and Cook, P. R., "A Meta-Instrument for Interactive, On-the-Fly Machine Learning.," in *NIME*, pp. 280–285, 2009.

[41] Fishkin, K. P., Moran, T. P., and Harrison, B. L., "Embodied user interfaces: Towards invisible user interfaces," in *Engineering for Human-Computer Interaction*, pp. 1–18, Springer, 1999.

[42] Fu, A. W.-C., Keogh, E., Lau, L. Y., Ratanamahatana, C. A., and Wong, R. C.-W., "Scaling and time warping in time series querying," *VLDB Journal*, vol. 17, no. 4, pp. 899–921, 2008.

[43] Gillian, N. and Paradiso, J. A., "The Gesture Recognition Toolkit," *JMLR*, vol. 15, pp. 3483–3487, Jan. 2014.

[44] Goel, M., Jansen, A., Mandel, T., Patel, S. N., and Wobbrock, J. O., "ContextType: Using Hand Posture Information to Improve Mobile Touch Screen Text Entry," in *Proc. CHI*, pp. 2795–2798, ACM, 2013.

[45] Goel, M., Wobbrock, J., and Patel, S., "GripSense: Using Built-in Sensors to Detect Hand Posture and Pressure on Commodity Mobile Phones," in *Proc. UIST*, pp. 545–554, ACM, 2012.

[46] Gouveia, R., Pereira, F., Karapanos, E., Munson, S. A., and Hassenzahl, M., "Exploring the Design Space of Glanceable Feedback for Physical Activity Trackers," in *Proc. Ubicomp*, pp. 144–155, ACM, 2016.

[47] Greenberg, S. and Buxton, B., "Usability evaluation considered harmful (some of the time)," in *Proc. CHI*, pp. 111–120, ACM, 2008.

[48] Greenberg, S. and Fitchett, C., "Phidgets: Easy Development of Physical Interfaces Through Physical Widgets," in *Proc. UIST*, pp. 209–218, ACM, 2001.

[49] Gupta, S., Morris, D., Patel, S., and Tan, D., "SoundWave: Using the Doppler Effect to Sense Gestures," in *Proc. CHI*, pp. 1911–1914, 2012.

[50] Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., and Witten, I. H., "The WEKA data mining software: an update," *SIGKDD*, pp. 10–18, 2009.

[51] Hammerla, N. Y., Kirkham, R., Andras, P., and Ploetz, T., "On preserving statistical characteristics of accelerometry data using their empirical cumulative distribution," in *Proc. ISWC*, p. 65, 2013.

[52] Hartmann, B., Abdulla, L., Mittal, M., and Klemmer, S. R., "Authoring Sensor-based Interactions by Demonstration with Direct Manipulation and Pattern Recognition," in *Proc. CHI*, pp. 145–154, 2007.

[53] Hightower, J. and Borriello, G., "Location systems for ubiquitous computing," *Computer*, vol. 34, no. 8, pp. 57–66, 2001.

[54] Hinckley, K., "Synchronous Gestures for Multiple Persons and Computers," in *Proc. UIST*, pp. 149–158, ACM, 2003.

[55] Hinckley, K., Chen, X. A., and Benko, H., "Motion and Context Sensing Techniques for Pen Computing," in *Proc. GI*, GI '13, pp. 71–78, Canadian Information Processing Society, 2013.

[56] Hinckley, K., Pahud, M., Benko, H., Irani, P., Guimbretiere, F., Gavriliu, M., Chen, X. A., Matulic, F., Buxton, W., and Wilson, A., "Sensing Techniques for Tablet+Stylus Interaction," in *Proc. UIST*, pp. 605–614, 2014.

[57] Hinckley, K., Pausch, R., Goble, J. C., and Kassell, N. F., "A Survey of Design Issues in Spatial Input," in *Proc. UIST*, pp. 213–222, ACM, 1994.

[58] Hinckley, K., Pierce, J., Sinclair, M., and Horvitz, E., "Sensing techniques for mobile interaction," in *Proc. UIST*, pp. 91–100, 2000.

[59] Hinckley, K. and Song, H., "Sensor synaesthesia: touch in motion, and motion in touch," in *Proc. CHI*, pp. 801–810, 2011.

[60] Hinckley, K. and Wigdor, D., "Input technologies and techniques," *The human-computer interaction handbook: fundamentals, evolving technologies and emerging applications*, pp. 151–168, 2002.

[61] Hinckley, K., Yatani, K., Pahud, M., Coddington, N., Rodenhouse, J., Wilson, A., Benko, H., and Buxton, B., "Pen+ touch= new tools," in *Proc. UIST*, pp. 27–36, 2010.

[62] Houben, S. and Marquardt, N., "WatchConnect: A Toolkit for Prototyping Smartwatch-Centric Cross-Device Applications," in *Proc. CHI*, pp. 1247–1256, ACM, 2015.

[63] Hudson, S. E. and Mankoff, J., "Concepts, Values, and Methods for Technical Human-Computer Interaction Research," in *Ways of Knowing in HCI*, pp. 69–93, 2014.

[64] Hutchins, E., Hollan, J., and Norman, D., "Direct Manipulation Interfaces," *Human-Computer Interaction*, vol. 1, pp. 311–338, Dec. 1985.

[65] Hkkil, J., Rantakari, J., Roinesalo, P., and Colley, A., "Charting User Preferences on Wearable Visual Markers," in *Proc. ISWC*, pp. 60–61, ACM, 2016.

[66] Jacob, R. J., Girouard, A., Hirshfield, L. M., Horn, M. S., Shaer, O., Solovey, E. T., and Zigelbaum, J., "Reality-based Interaction: A Framework for post-WIMP Interfaces," in *Proc. CHI*, CHI '08, pp. 201–210, ACM, 2008.

[67] Kamal, A., Li, Y., and Lank, E., "Teaching motion gestures via recognizer feedback," in *Proc. IUI*, pp. 73–82, 2014.

[68] Kim, J.-W., Kim, H.-J., and Nam, T.-J., "M.Gesture: An Acceleration-Based Gesture Authoring System on Multiple Handheld and Wearable Devices," in *Proc. CHI*, pp. 2307–2318, 2016.

[69] Kim, J.-W. and Nam, T.-J., "EventHurdle: Supporting Designers' Exploratory Interaction Prototyping with Gesture-based Sensors," in *Proc. CHI*, pp. 267–276, 2013.

[70] Kim, J.-W., Nam, T.-J., and Park, T., "CompositeGesture: Creating Custom Gesture Interfaces with Multiple Mobile or Wearable Devices," *International Journal on Interactive Design and Manufacturing (IJIDeM)*, pp. 1–6, Jan. 2014.

[71] Kin, K., Hartmann, B., DeRose, T., and Agrawala, M., "Proton++: a customizable declarative multitouch framework," in *Proc. UIST*, pp. 477–486, ACM, 2012.

[72] Kin, K., Hartmann, B., DeRose, T., and Agrawala, M., "Proton: multitouch gestures as regular expressions," in *Proc. CHI*, pp. 2885–2894, ACM, 2012.

[73] Klemmer, S. R., Hartmann, B., and Takayama, L., "How Bodies Matter: Five Themes for Interaction Design," in *Proc. DIS*, pp. 140–149, 2006.

[74] Klemmer, S. R., Sinha, A. K., Chen, J., Landay, J. A., Aboobaker, N., and Wang, A., "Suede: a Wizard of Oz prototyping tool for speech user interfaces," in *Proc. UIST*, pp. 1–10, 2000.

[75] Kohlsdorf, D. K. H. and Starner, T. E., "MAGIC summoning: towards automatic suggesting and testing of gestures with low probability of false positives during use," *JMLR*, vol. 14, no. 1, pp. 209–242, 2013.

[76] Kratz, S. and Rohs, M., “A $3 gesture recognizer: simple gesture recognition for devices equipped with 3d acceleration sensors,” in *Proc. IUI*, pp. 341–344, ACM, 2010.

[77] Kratz, S. and Rohs, M., “Protractor3d: A Closed-form Solution to Rotation-invariant 3d Gestures,” in *Proc. IUI*, IUI ’11, pp. 371–374, ACM, 2011.

[78] Kray, C., Nesbitt, D., Dawson, J., and Rohs, M., “User-defined Gestures for Connecting Mobile Phones, Public Displays, and Tabletops,” in *Proc. MobileHCI*, pp. 239–248, ACM, 2010.

[79] Krumm, J., *Ubiquitous computing fundamentals.* Boca Raton: Chapman & Hall/CRC Press, 2010.

[80] Kwapisz, J. R., Weiss, G. M., and Moore, S. A., “Activity Recognition Using Cell Phone Accelerometers,” *SIGKDD*, vol. 12, no. 2, pp. 74–82, 2011.

[81] Landay, J. A. and Myers, B. A., “Interactive sketching for the early stages of user interface design,” in *Proc. CHI*, pp. 43–50, ACM Press/Addison-Wesley Publishing Co., 1995.

[82] Landay, J. A. and Myers, B. A., “Sketching interfaces: Toward more human interface design,” *Computer*, vol. 34, no. 3, pp. 56–64, 2001.

[83] Lau, T., *Programming by demonstration: a machine learning approach.* PhD thesis, University of Washington, 2001.

[84] LaViola, Jr., J. J., “Context Aware 3d Gesture Recognition for Games and Virtual Reality,” in *ACM SIGGRAPH 2015 Courses*, SIGGRAPH ’15, (New York, NY, USA), pp. 10:1–10:61, ACM, 2015.

[85] Li, Y., “Protractor: A Fast and Accurate Gesture Recognizer,” in *Proc. CHI*, CHI ’10, pp. 2169–2172, ACM, 2010.

[86] Li, Y. and Landay, J. A., “Activity-based Prototyping of Ubicomp Applications for Long-lived, Everyday Human Activities,” in *Proc. CHI*, pp. 1303–1312, ACM, 2008.

[87] Lim, B. Y. and Dey, A. K., “Toolkit to support intelligibility in context-aware applications,” in *Proceedings of the 12th ACM international conference on Ubiquitous computing*, pp. 13–22, ACM, 2010.

[88] Liu, P., Chen, Y., Tang, W., and Yue, Q., “Mobile WEKA as Data Mining Tool on Android,” in *EEA*, no. 139 in Advances in Intelligent and Soft Computing, pp. 75–80, Jan. 2012.

[89] Long, A. C., Landay, J. A., and Rowe, L. A., “Those look similar! Issues in automating gesture design advice,” in *Proc. PUI*, pp. 1–5, 2001.

[90] Long, Jr., A. C., Landay, J. A., and Rowe, L. A., “Implications for a Gesture Design Tool,” in *Proc. CHI*, pp. 40–47, ACM, 1999.

[91] Long Jr, A. C., *Quill: a gesture design tool for pen-based user interfaces.* PhD thesis, University of California, Berkeley, 2001.

[92] Lu, C.-L. and Soo, V.-W., "Context-dependent action interpretation in interactive storytelling games," in *Proc. ACHI*, pp. 7–10, ACM, 2012.

[93] Lu, H. and Li, Y., "Gesture Coder: A Tool for Programming Multi-touch Gestures by Demonstration," in *Proc. CHI*, pp. 2875–2884, ACM, 2012.

[94] Lu, H. and Li, Y., "Gesture Studio: Authoring Multi-touch Interactions Through Demonstration and Declaration," in *Proc. CHI*, pp. 257–266, ACM, 2013.

[95] Lyons, K., Brashear, H., Westeyn, T., Kim, J. S., and Starner, T., "Gart: The gesture and activity recognition toolkit," in *Human-Computer Interaction. HCI Intelligent Multimodal Interaction Environments*, pp. 718–727, Springer, 2007.

[96] Mistry, P. and Maes, P., "SixthSense: a wearable gestural interface," in *SIGGRAPH Sketches*, p. 11, 2009.

[97] Montero, C. S., Alexander, J., Marshall, M. T., and Subramanian, S., "Would you do that?: understanding social acceptance of gestural interfaces," in *Proc. MobileHCI*, pp. 275–278, ACM, 2010.

[98] Myers, B., Hudson, S. E., and Pausch, R., "Past, present, and future of user interface software tools," *ACM TOCHI*, vol. 7, no. 1, pp. 3–28, 2000.

[99] Myers, B. A., "User Interface Software Tools," *ACM TOCHI*, vol. 2, no. 1, pp. 64–103, 1995.

[100] Myers, B. A., McDaniel, R. G., and Kosbie, D. S., "Marquise: Creating Complete User Interfaces by Demonstration," in *Proc. CHI*, pp. 293–300, ACM, 1993.

[101] Negulescu, M., Ruiz, J., Li, Y., and Lank, E., "Tap, Swipe, or Move: Attentional Demands for Distracted Smartphone Input," in *Proc. AVI*, pp. 173–180, 2012.

[102] Nigay, L. and Coutaz, J., "A Design Space for Multimodal Systems: Concurrent Processing and Data Fusion," in *Proc. CHI*, pp. 172–178, ACM, 1993.

[103] Norman, D. A. and Nielsen, J., "Gestural Interfaces: A Step Backward in Usability," *interactions*, vol. 17, pp. 46–49, Sept. 2010.

[104] Olsen Jr, D. R., "Evaluating user interface systems research," in *Proc. UIST*, pp. 251–258, 2007.

[105] O'Sullivan, D. and Igoe, T., *Physical computing: sensing and controlling the physical world with computers.* Course Technology Press, 2004.

[106] Parnami, A., Gupta, A., Reyes, G., Sadana, R., Li, Y., and Abowd, G., "Mogeste: Mobile Tool for In-situ Motion Gesture Design," in *Proc. UbiComp Adjunct*, pp. 345–348, ACM, 2016.

[107] Parnami, A., Gupta, A., Reyes, G., Sadana, R., Li, Y., and Abowd, G. D., "Mogeste: A Mobile Tool for In-Situ Motion Gesture Design," in *Proc. IHCI*, pp. 35–43, ACM, 2016.

[108] Patel, K., Fogarty, J., Landay, J. A., and Harrison, B., "Investigating Statistical Machine Learning As a Tool for Software Development," in *Proc. CHI*, pp. 667–676, 2008.

[109] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., and others, "Scikit-learn: Machine learning in Python," *JMLR*, vol. 12, pp. 2825–2830, 2011.

[110] Rabiner, L., "A tutorial on hidden Markov models and selected applications in speech recognition," *Proc. IEEE*, vol. 77, no. 2, pp. 257–286, 1989.

[111] Rekimoto, J., "Gesturewrist and gesturepad: Unobtrusive wearable interaction devices," in *Proc. ISWC*, pp. 21–27, 2001.

[112] Reyes, G., Zhang, D., Ghosh, S., Shah, P., Wu, J., Parnami, A., Bercik, B., Starner, T., Abowd, G. D., and Edwards, W. K., "Whoosh: Non-voice Acoustics for Low-cost, Hands-free, and Rapid Input on Smartwatches," in *Proc. ISWC*, ISWC '16, pp. 120–127, ACM, 2016.

[113] Rogers, Y., Sharp, H., and Preece, J., *Interaction Design: Beyond Human - Computer Interaction*. Interaction Design: Beyond Human-computer Interaction, Wiley, 2011.

[114] Rubine, D., *Specifying gestures by example*, vol. 25. 1991.

[115] Ruiz, J. and Li, Y., "DoubleFlip: A Motion Gesture Delimiter for Mobile Interaction," in *Proc. CHI*, pp. 2717–2720, ACM, 2011.

[116] Ruiz, J., Li, Y., and Lank, E., "User-defined Motion Gestures for Mobile Interaction," in *Proc. CHI*, pp. 197–206, 2011.

[117] Salber, D., Dey, A. K., and Abowd, G. D., "The context toolkit: aiding the development of context-enabled applications," in *Proc. CHI*, pp. 434–441, ACM, 1999.

[118] Saponas, T. S., Tan, D. S., Morris, D., Balakrishnan, R., Turner, J., and Landay, J. A., "Enabling Always-available Input with Muscle-computer Interfaces," in *Proc. UIST*, UIST '09, (New York, NY, USA), pp. 167–176, ACM, 2009.

[119] Schlomer, T., Poppinga, B., Henze, N., and Boll, S., "Gesture recognition with a Wii controller," in *Proc. TEI*, pp. 11–14, 2008.

[120] Scholliers, C., Hoste, L., Signer, B., and De Meuter, W., "Midas: A Declarative Multi-touch Interaction Framework," in *Proc. TEI*, pp. 49–56, ACM, 2011.

[121] Scott, J., Dearman, D., Yatani, K., and Truong, K. N., "Sensing Foot Gestures from the Pocket," in *Proc. UIST*, pp. 199–208, ACM, 2010.

[122] Seifi, H. and Lyons, K., "Exploring the Design Space of Touch-based Vibrotactile Interactions for Smartwatches," in *Proc. ISWC*, pp. 156–165, ACM, 2016.

[123] Selker, T. and Burleson, W., "Context-aware design and interaction in computer systems," *IBM Systems Journal*, vol. 39, no. 3.4, pp. 880–891, 2000.

[124] Song, J., Soros, G., Pece, F., Fanello, S. R., Izadi, S., Keskin, C., and Hilliges, O., "In-air Gestures Around Unmodified Mobile Devices," in *Proc. UIST*, pp. 319–329, 2014.

[125] Starner, T., "The challenges of wearable computing: Part 2," *Micro, IEEE*, vol. 21, no. 4, pp. 54–67, 2001.

[126] Starner, T., Auxier, J., Ashbrook, D., and Gandy, M., "The gesture pendant: A self-illuminating, wearable, infrared computer vision system for home automation control and medical monitoring," in *Proc. ISWC*, pp. 87–94, 2000.

[127] Sutherland, I. E., "Sketch pad a man-machine graphical communication system," in *Proceedings of the SHARE design automation workshop*, pp. 6–329, ACM, 1964.

[128] Valentin, G., Alcaidinho, J., Howard, A., Jackson, M. M., and Starner, T., "Towards a Canine-human Communication System Based on Head Gestures," in *Proc. ACE*, pp. 65:1–65:9, ACM, 2015.

[129] Ward, J. A., Lukowicz, P., and Gellersen, H. W., "Performance metrics for activity recognition," *ACM TIST*, vol. 2, no. 1, p. 6, 2011.

[130] Wen, H., Ramos Rojas, J., and Dey, A. K., "Serendipity: Finger Gesture Recognition Using an Off-the-Shelf Smartwatch," in *Proc. CHI*, pp. 3847–3851, ACM, 2016.

[131] Westeyn, T., Brashear, H., Atrash, A., and Starner, T., "Georgia Tech Gesture Toolkit: Supporting Experiments in Gesture Recognition," in *Proc. ICMI*, ICMI '03, pp. 85–92, ACM, 2003.

[132] Wexelblat, A., "Research challenges in gesture: Open issues and unsolved problems," in *Gesture and sign language in human-computer interaction*, pp. 1–11, Springer, 1997.

[133] Wobbrock, J. O., Morris, M. R., and Wilson, A. D., "User-defined gestures for surface computing," in *Proc. CHI*, pp. 1083–1092, 2009.

[134] Wobbrock, J. O., Wilson, A. D., and Li, Y., "Gestures Without Libraries, Toolkits or Training: A $1 Recognizer for User Interface Prototypes," in *Proc. UIST*, UIST '07, (New York, NY, USA), pp. 159–168, ACM, 2007.

[135] Wu, J., Pan, G., Zhang, D., Qi, G., and Li, S., "Gesture Recognition with a 3-D Accelerometer," in *UIC* (Zhang, D., Portmann, M., Tan, A.-H., and Indulska, J., eds.), no. 5585 in Lecture Notes in Computer Science, pp. 25–38, 2009.

[136] Zamborlin, B., Bevilacqua, F., Gillies, M., and D'inverno, M., "Fluid Gesture Interaction Design: Applications of Continuous Recognition for the Design of Modern Gestural Interfaces," *ACM TiiS*, vol. 3, pp. 22:1–22:30, Jan. 2014.

[137] Zhang, C., Bedri, A., Reyes, G., Bercik, B., Inan, O. T., Starner, T. E., and Abowd, G. D., "TapSkin: Recognizing On-Skin Input for Smartwatches," in *Proc. ISS*, pp. 13–22, ACM, 2016.

[138] Zhang, C., Guo, A., Zhang, D., Southern, C., Arriaga, R., and Abowd, G., "BeyondTouch: Extending the Input Language with Built-in Sensors on Commodity Smartphones," in *Proc. IUI*, pp. 67–77, 2015.

[139] Zhang, C., Parnami, A., Southern, C., Thomaz, E., Reyes, G., Arriaga, R., and Abowd, G. D., "BackTap: Robust Four-point Tapping on the Back of an Off-the-shelf Smartphone," in *Proc. UIST Adjunct*, pp. 111–112, ACM, 2013.

[140] Zhang, C., Yang, J., Southern, C., Starner, T. E., and Abowd, G. D., "WatchOut: Extending Interactions on a Smartwatch with Inertial Sensing," in *Proc. ISWC*, pp. 136–143, ACM, 2016.